

UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE CIENCIA
Departamento de Matemática y Ciencia de la Computación



Diseño Inverso de cavidades Fabry-Perot utilizando redes neuronales

Bastián Martínez

Profesor guía: Felipe A. Herrera Urbina

Trabajo de Titulación para optar al título de Analista en
Computación Científica.

Santiago - Chile
2023

Resumen

El desarrollo de nuevos materiales y estructuras en campos como la nanofotónica y la ciencia de materiales requiere métodos eficientes y precisos. Tradicionalmente, este proceso ha sido dominado por enfoques experimentales y numéricos que, aunque efectivos, pueden ser extremadamente lentos y costosos. En este contexto, el diseño inverso emerge como una alternativa, permitiendo la obtención de estructuras a partir de las funciones deseadas. Este trabajo se centra en la utilización de redes neuronales para el diseño inverso de cavidades Fabry-Pérot (F-P).

Se desarrolló un modelo basado en redes neuronales que aprende la relación entre los espectros de transmisión óptica y la configuración física de las cavidades F-P. El modelo se entrenó y validó utilizando un conjunto de datos generado mediante el método de matrices de transferencia. Los resultados muestran que el modelo es capaz de estimar las estructuras de nuevas cavidades a partir de un espectro de transmisión dado, con un error cuadrático medio (ECM) y un error absoluto medio (EAM) bajos en el conjunto de prueba.

A pesar de la capacidad de la arquitectura utilizada, se identificaron limitaciones relacionadas con la generalización del modelo a diferentes tipos de materiales y configuraciones de cavidad. Estas restricciones sugieren áreas para futuras investigaciones, incluyendo el desarrollo de modelos más flexibles capaces de adaptarse a más materiales y estructuras. Este trabajo confirma la viabilidad del diseño inverso mediante inteligencia artificial para cavidades F-P y entrega una herramienta alternativa a los métodos numéricos tradicionales.

Palabras clave: Diseño inverso, redes neuronales, inteligencia artificial, nanofotónica, ciencia de materiales.

Abstract

The development of new materials and structures in fields such as nanophotonics and material science requires efficient and precise methods. Traditionally, this process has been dominated by experimental and numerical approaches which, although effective, can be extremely slow and costly. In this context, inverse design emerges as an alternative, enabling the derivation of structures from desired functions. This work focuses on the use of neural networks for the inverse design of Fabry-Pérot (F-P) cavities.

A neural network-based model was developed to learn the relationship between optical transmission spectra and the physical configuration of F-P cavities. The model was trained and validated using a dataset generated through the transfer matrix method. The results show that the model is capable of estimating the structures of new cavities from a given transmission spectrum, with low mean squared error (MSE) and mean absolute error (MAE) on the test dataset.

Despite the capability of the architecture used, limitations related to the model's generalization to different types of materials and cavity configurations were identified. These limitations suggest areas for future research, including the development of more flexible models capable of adapting to more materials and structures. This work confirms the viability of inverse design through artificial intelligence for F-P cavities and provides an alternative tool to traditional numerical methods.

Key words: Inverse design, neural networks, artificial intelligence, nanophotonics, materials science.

Dedicatoria

A mi madre Susana y mi padre Victor

Agradecimientos

Quiero agradecer al profesor Felipe Herrera quien me ha guiado y ayudado en el desarrollo de esta tesis y además, por haberme dado la oportunidad de conocer el mundo de la investigación científica.

Agradecer a mi familia, amigos y todas las personas que han estado conmigo desde que tengo memoria, en especial a Ámbar quien su apoyo y compañía ha sido realmente importante.

Un especial agradecimiento al profesor Rubén Muñoz quien, en el colegio Antú de la Cisterna en los cursos 7° B y 8° B nos enseñó con esmero la responsabilidad, el respetar a los demás, ser buenas personas y por sobre todo, nunca rendirse.

Índice general

Índice de ilustraciones	II
Introducción	1
Objetivo general	3
1. Marco teórico	4
1.1. Justificación y estado del arte	4
1.2. Conceptos fundamentales	6
1.2.1. Diseño inverso	6
1.2.2. Cavidades Fabry-Perot	7
1.2.3. Inteligencia artificial	9
1.2.4. Aprendizaje supervisado y no supervisado	12
1.2.5. Redes neuronales	14
1.3. Optimizador	22
1.4. Técnicas de Regularización	24
1.5. Descripción de la problemática	25
1.6. Hipótesis y postulados	27
2. Metodología	28
2.1. Diseño de la Investigación	28
2.2. Herramientas y Tecnologías	29
2.3. Datos	29
2.3.1. Exploración de datos	30
2.4. Modelo de red neuronal	33
2.5. Entrenamiento red neuronal	33
2.5.1. Entrenamiento red neuronal	35
3. Resultados	37
3.1. Discusión	40
3.2. Conclusiones	41

Índice de ilustraciones

1.1. Esquema de diseño clásico e inverso. Elaboración propia	7
1.2. Esquema de cavidad Fabry-Perot. Elaboración propia	9
1.3. Esquema de inteligencia artificial, aprendizaje automático y profundo. Elaboración propia	12
1.4. Tipos de aprendizaje en ML. Elaboración propia	14
1.5. Esquema de un Perceptrón. Elaboración propia	15
1.6. Funciones de activación. Elaboración propia	17
1.7. Red Neuronal Básica. Elaboración propia	19
1.8. Cavidad F-P utilizada. Elaboración propia	25
1.9. Espectro TRA para una cavidad F-P. Elaboración propia	26
2.1. Distribución de espesores de plata y silica en los datos de entrenamiento	31
2.2. Diferencia de espesores de plata y silica	32
2.3. Distribución de datos de TDBC	32
2.4. Arquitectura de red neuronal	34
2.5. MSE para los K-Folds en red RITE	35
3.1. Comparación método matriz de transferencia y red neuronal	37
3.2. Diagrama de flujo de graficar espectros T	38
3.3. Distribución de errores por material en la cavidad F-P. Los histogramas muestran la frecuencia de los errores de predicción.	40

Introducción

La problemática a abordar en el trabajo se centra en el diseño de cavidades Fabry-Perot (FP) mediante redes neuronales. Estas cavidades son esenciales en diversas aplicaciones tecnológicas, como la detección, la emisión de radiación, la comunicación óptica y la espectroscopia (Saleh & Teich, 2007). Actualmente, el proceso de diseño y optimización de estas cavidades es costoso en tiempo y recursos monetarios y a menudo basado en métodos heurísticos de ensayo-error.

Esta problemática se enfrenta utilizando técnicas de optimización como los algoritmos evolutivos (Bäck et al., 2005), métodos de búsqueda local y optimización global. Según (Liu et al., 2019) el enfoque clásico comienza por una “apuesta” empírica de un investigador experto en el área seguido por un refinamiento iterativo de las propiedades y geometría específica del material. Con este enfoque existen inconvenientes como la lentitud, la tendencia de los métodos de optimización a quedarse atrapados en mínimos locales y la dificultad para explorar y explotar eficientemente el espacio de búsqueda debido a la gran cantidad de grados de libertad e interrelaciones de la física propia del problema.

Según (Molesky et al., 2018) el diseño inverso se propone como una solución no tradicional al problema de diseño debido la óptica no-lineal que tienen los desafíos en fotónica a escala nano. El enfoque del diseño inverso, en el que se buscan materiales o funcionalidades con resultados específicos, en lugar de determinar las propiedades de un material o configuración dada, ha demostrado ser una herramienta poderosa en el campo de la ciencia de materiales (Zunger, 2018).

El término Machine Learning (ML) definido por primera vez en (Kohavi & Provost, 1998) hace referencia a la capacidad de un algoritmo de “aprender” a solucionar el problema para el que fue diseñado. El ML tiene la capacidad de resolver problemas de alta dimensionalidad por lo que ha ganado gran popularidad desde sus inicios. En los últimos años se han hecho avances prometedores en materia de generación de muestras fuera del conjunto de entrenamiento propiciado por las redes generativas adversarias (GANs) propuestas por (Goodfellow et al., 2014) como un enfoque novedoso para el cumplimiento de tareas en donde se busca obtener nueva información a partir de información ya presente. En el trabajo de (Peurifoy et al., 2018) se demostró la efectividad de las redes neuronales en la simulación de partículas nano fotónicas en el mismo formato de capas tomando

como foco el diseño inverso de estas estructuras a escala nano. Por otra parte, se han realizado avances en las llamadas redes neuronales inversas para la generación guiada de imágenes (Ardizzone et al., 2019) como en problemas de propósito general (Ardizzone et al., 2018)), este tipo de redes convergen en menos tiempo y por su naturaleza son biyectivas, es decir, no hay que realizar más esfuerzo para invertir el problema.

Este trabajo se desarrollará un enfoque basado en redes neuronales para generar diseños de cavidades FP que cumplan con requisitos específicos y se espera un rendimiento superior al de métodos alternativos. El diseño inverso no sólo se puede utilizar con redes neuronales, sino que también es ampliado a problemas como diseño de metamateriales (Zangeneh-Nejad et al., 2020), en general se puede aplicar a problemas que sean sobre la búsqueda partir del resultado.

Además, veremos cómo solucionar el problema base de prueba de concepto de diseño inverso con redes neuronales y después veremos la generalización de la problemática y analizando la solución ante el problema de N a 1.

La contribución esperada de este trabajo al ámbito de la Ciencia de la Computación es la aplicación de redes neuronales al diseño inverso de cavidades F-P, proporcionando una metodología e implementación para abordar este problema. Se espera que esta investigación ayude a acelerar el proceso de diseño y optimización, reducir los costos asociados y generar soluciones más innovadoras. Finalmente, la solución quedará disponible en formato Open-Source.

Objetivo general

Desarrollar una herramienta computacional que facilite la búsqueda de cavidades F-P con una respuesta óptica de interés.

Objetivos específicos

- Comprender el funcionamiento de las cavidades F-P, en el sentido de cómo es la física y su incidencia en el problema.
- Investigar sobre estudios realizados en el campo de diseño inverso, en específico utilizando redes neuronales.
- Desarrollar y entrenar una red neuronal para la optimización y el diseño inverso de cavidades F-P.
- Evaluar, validar y comparar el rendimiento de las redes neuronales entrenados utilizando métricas cuantitativas.

Capítulo 1

Marco teórico

En este marco teórico se propone profundizar en los conceptos fundamentales relacionados con el diseño inverso, las cavidades F-P y las redes neuronales, ilustrando cómo estos campos se interrelacionan. El objetivo de esta sección es establecer un entendimiento común y proporcionar un contexto relevante que permita abordar de manera efectiva el contenido presentado. En la construcción de este marco, se buscará integrar literatura académica y técnica, que servirá como base para las reflexiones posteriores y el análisis de los resultados obtenidos.

1.1. Justificación y estado del arte

El descubrimiento de nuevos materiales ha sido de sumo interés para la humanidad desde la prehistoria con la edad de piedra y el uso de herramientas rústicas, luego la edad del hierro en donde se desarrollaron mejoras en las herramientas de piedra a través de la forja, a los tiempos actuales en donde la ciencia de materiales ha avanzado a tal punto que podemos manipular materiales con suma exactitud. El descubrimiento de nuevos materiales para suplir las necesidades y requerimientos de la era actual cada vez se vuelve más complicado ya que los métodos convencionales no son lo suficientemente eficientes. Según Liu et al. (2017) el proceso de descubrimiento experimental de un nuevo material toma aproximadamente 10 a 20 años desde su investigación hasta su lanzamiento en la industria. El proceso convencional de diseño de materiales que consta de siete fases: Descubrimiento, Desarrollo, Optimización, Diseño, Certificación, Manufactura y Entrega, este proceso suele ser muy costoso en términos monetarios ya que se deben realizar pruebas de validación y seguridad constantemente. Un enfoque alternativo al experimental es el de la simulación de las interacciones entre elementos para crear nuevos materiales, sin embargo esta alternativa no está libre de problemas, por ejemplo la alta capacidad de cómputo requerido para realizar simulaciones a nivel molecular de las interacciones entre elementos. Por lo tanto, la búsqueda de métodos eficientes para el descubrimiento

de materiales, que permitan reducir costes tanto en tiempo como en recursos económicos, se ha vuelto imprescindible. En la actualidad, se observa un creciente interés en abordar esta problemática y en desarrollar un marco de trabajo específico para el descubrimiento de nuevos materiales. Uno de los enfoques en auge es el uso de la inteligencia artificial, la cual permite modelar la complejidad de las interacciones que son difíciles de analizar mediante métodos numéricos o que exigen una gran capacidad de cálculo.

En los últimos años, se ha observado un incremento en la publicación de artículos científicos enfocados en el diseño inverso mediante el uso de redes neuronales. Este auge se debe a los progresos alcanzados en el campo, así como la eficiencia que se alcanza con este enfoque. Por ejemplo, investigaciones como la realizada en (Molesky et al., 2018), que destaca cómo, en el ámbito de la nanofotónica, la complejidad de optimizar múltiples variables de manera simultánea hace que los métodos convencionales de diseño fotónico queden en desventaja. En contraste, los enfoques de diseño inverso emergen como soluciones preferentes para el estudio y desarrollo de nuevas estructuras. Específicamente, se resalta la ventaja de emplear algoritmos capaces de deducir las propiedades físicas deseadas a partir de una descripción funcional del sistema. En otras palabras, estos métodos permiten definir la función que se espera del material para, a partir de ahí, determinar la composición necesaria para su fabricación.

Por otro lado (Liu et al., 2018) se examina la aplicación de redes neuronales para el diseño de metasuperficies, utilizando una versión modificada de redes generativas adversarias (GAN) en donde a la esquema original se le agrega una tercera red que llaman Simulador guía al Generador basándose en los parámetros específicos de la metasuperficie que se pretende obtener. Los resultados indican que este enfoque ofrece una alternativa al método de Modelado por Elementos Finitos (FEM), utilizado para comparación. Además, los autores señalan la capacidad de este método para generar nuevas estructuras de manera eficiente.

También en (Kim et al., 2021) se aborda el diseño inverso de OLEDs empleando redes neuronales en tándem, una arquitectura que involucra dos redes neuronales operando en secuencia, siendo una la inversa de la otra. Según los autores, la simulación llevada a cabo mediante redes neuronales demuestra ser hasta tres órdenes de magnitud más eficiente en comparación con las simulaciones electromagnéticas tradicionales del mismo problema.

La diversidad de métodos disponibles para llevar a cabo una tarea resulta crucial en el ámbito de la investigación, ya que facilita el contraste y la comparación de resultados. La importancia de esta investigación radica en su contribución de un nuevo método que emplea redes neuronales, avanzando así en el conocimiento sobre alternativas para la exploración de cavidades Fabry-Pérot (F-P). Este enfoque promueve además la colaboración interdisciplinaria entre la Ciencia de la Computación y la Física.

1.2. Conceptos fundamentales

1.2.1. Diseño inverso

El diseño clásico se puede describir como el proceso en que, para resolver un problema, es necesario seguir una serie de pasos preliminares. Este proceso se orienta fundamentalmente en una secuencia lineal de acciones: definir el problema, generar un modelo, simular el comportamiento del modelo y, por último, verificar si el resultado cumple con los objetivos planteados.

Para ilustrar este concepto, consideremos una función $f : X \rightarrow Y$, donde X es el conjunto de entradas (*input*) y Y el conjunto de salidas (*output*). En el diseño clásico, partimos de un punto en X para encontrar su correspondiente punto en Y mediante la aplicación de f . Sin embargo el encontrar el Y deseado no es trivial, esto requiere de un proceso que a menudo comienza con un X aleatorio, para luego evaluar la respuesta por medio de simulaciones y ajustar este X iterativamente en base a la comparación con la respuesta objetivo. Este proceso puede requerir cientos o miles de simulaciones, lo que, debido al alto costo computacional asociado, resulta extremadamente lento a medida que aumentan el tamaño y la complejidad del dispositivo.

El diseño inverso, por otro lado, invierte esta lógica. En lugar de partir de las entradas para obtener las salidas, se comienza con el resultado deseado y se trabaja hacia atrás para determinar las entradas necesarias. Esto implica utilizar la función inversa $f^{-1} : Y \rightarrow X$, de modo que partiendo de un punto en Y se busca su preimagen en X . Lo anterior significa que si se tiene un resultado específico que se desea lograr, el diseño inverso identifica el conjunto de parámetros o condiciones iniciales que, al aplicar el proceso o función inversa, producen el resultado deseado

Dentro del campo de estudio, se han desarrollado investigaciones que siguen el enfoque de diseño clásico. Por ejemplo, el descrito por matrices de transferencia, visto en (Abelardo et al., 2019), que utiliza para simular el comportamiento de sistemas OLED ante la transmisión de fotones dentro del dispositivo. Asimismo, el método Chance-Prock-Silbey (CPS), que también se fundamenta en el uso de matrices de transferencia, ha sido explorado por (Kim et al., 2021). Una limitación inherente a estos enfoques de diseño clásico es su alta sensibilidad a modificaciones estructurales del sistema.

El diseño inverso se presenta como la inversión del proceso descrito previamente. Este enfoque altera la secuencia de operaciones, comenzando con el resultado o comportamiento deseado y obteniendo los parámetros o configuraciones de entrada que producirían dicho resultado. Este enfoque facilita la identificación de parámetros, especialmente en sistemas complejos donde la relación entre la estructura y la función no es lineal. La diferencia entre ambos enfoques, se ilustra en la figura 1.1

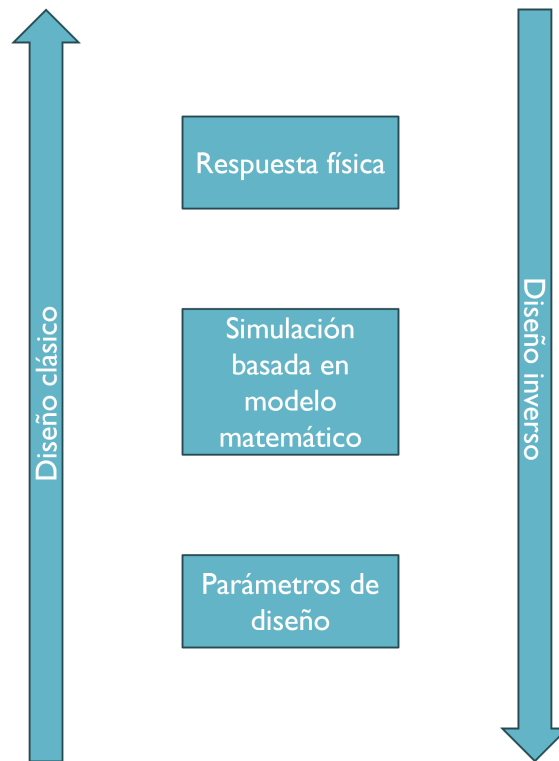


Figura 1.1: Esquema de diseño clásico e inverso. Elaboración propia

1.2.2. Cavidades Fabry-Perot

Una cavidad Fabry-Perot (F-P) es un resonador óptico compuesto por dos espejos paralelos y altamente reflectantes. La capacidad de las cavidades F-P para generar patrones de interferencia de alta resolución las hace útiles para aplicaciones que requieren una medición precisa de la longitud de onda de la luz, coherencia espectral y líneas espectrales. El principio de funcionamiento de una cavidad F-P se asienta sobre la interferencia constructiva y destructiva de ondas de luz que se reflejan múltiples veces dentro de la cavidad.

Su nombre Fabry-Perot viene de los físicos franceses Charles Fabry y Alfred Perot, quienes describieron y detallaron su funcionamiento a principios del siglo XX (Perot & Fabry, 1899), una de sus principales utilidades en física es el producir patrones de interferencia de alta resolución

Los espejos pueden estar separados por aire o por otros medios, estos son recubiertos para permitir un grado de reflexión dentro de la cavidad. Es necesario recordar que la luz se puede modelar como una onda electromagnética, cuando las ondas entran en la cavidad se reflejan múltiples veces en los espejos 1.8, cada rayo de luz puede tomar diferentes rutas (longitud de camino óptico) a través de la cavidad antes de salir, el resultado es un

patrón de interferencia que se forma ocurre debido a una superposición de ondas de luz. Cuando dos ondas se encuentran, su interacción puede ser constructiva o destructiva, dependiendo de su fase relativa. En la interferencia constructiva, los picos y valles de las ondas coinciden, reforzándose mutuamente y aumentando la intensidad de la luz. Por el contrario, en la interferencia destructiva, los picos de una onda coinciden con los valles de otra, cancelándose mutuamente y disminuyendo la intensidad de la luz.

La interacción de las ondas de luz con los espejos de la cavidad se describe mediante las ecuaciones de Fresnel, mientras que la interferencia resultante de las múltiples reflexiones se explica a través de la ecuación de interferencia de ondas:

$$I = I_0 \cos^2 \left(\frac{2\pi d \cos(\theta)}{\lambda} \right), \quad (1.1)$$

donde I es la intensidad de la luz, I_0 la intensidad inicial, d la separación entre los espejos, λ la longitud de onda de la luz incidente y θ el ángulo de incidencia. La condición para que ocurra la interferencia constructiva es que la diferencia de camino óptico entre los rayos reflejados sea un múltiplo entero de la longitud de onda, lo que conduce a la formación de franjas de luz.

Es importante mencionar que la interferencia solo se produce para ciertas longitudes de onda de la luz, las que satisfacen la condición de resonancia de la cavidad, por esta razón las cavidades F-P son a menudo utilizadas para filtrar longitudes de onda de la luz en el campo de la espectroscopia.

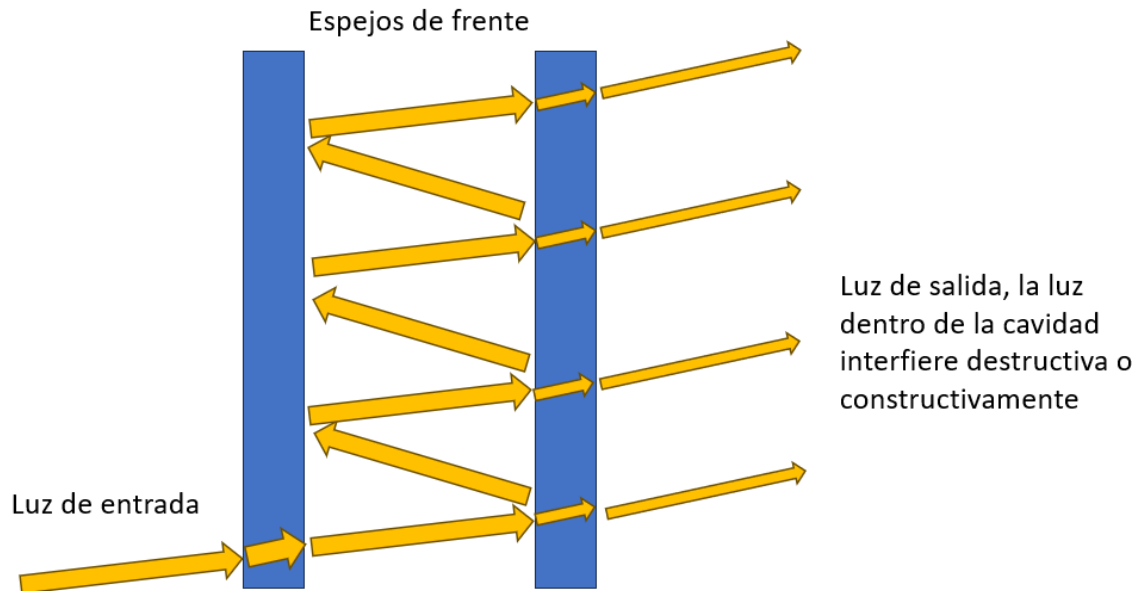


Figura 1.2: Esquema de cavidad Fabry-Perot. Elaboración propia

1.2.3. Inteligencia artificial

La Inteligencia Artificial (IA) ha experimentado una transformación significativa desde sus orígenes en la década de 1950. Una figura clave en este desarrollo fue Alan Turing, cuyo "Test de Turing" se ha convertido en un estándar de referencia en el campo. Este test propone un escenario en el que un juez humano interactúa con dos participantes: uno humano y otro una máquina. Si, tras estas interacciones, el juez es incapaz de distinguir cuál es el humano y cuál la máquina, se considera que la máquina ha demostrado inteligencia. Aunque existen debates sobre si el Test de Turing es un indicador definitivo de inteligencia en máquinas, es generalmente reconocido por plantear la fundamental cuestión de si las máquinas pueden pensar.

Un momento importante en la historia de la IA fue la Conferencia de Dartmouth de 1956, considerada el nacimiento de la IA como campo de estudio formal. Organizada por John McCarthy, Marvin Minsky, Nathaniel Rochester y Claude Shannon, esta conferencia reunió a investigadores para explorar cómo las máquinas podrían simular aspectos de la inteligencia humana. En su propuesta original, expresaron su convicción de que cada aspecto del aprendizaje o cualquier otra característica de la inteligencia humana podría ser descrito con tanta precisión que una máquina podría ser creada para simularlo.

Actualmente, la IA se clasifica en dos categorías: IA débil e IA fuerte. La IA débil se refiere a sistemas altamente especializados en resolver problemas acotados bajo condiciones determinadas, a menudo con mayor eficiencia y rapidez que los humanos, pero sin

conciencia del trabajo que realizan. Por otro lado, la IA fuerte se relaciona con sistemas que poseen conciencia de las tareas que realizan, a menudo denominados como IA de propósito general. A la fecha, todas las aplicaciones de IA que conocemos y utilizamos corresponden a IA débiles y especializadas.

La tabla 1.1 ilustra las diferencias clave entre la IA débil y la IA fuerte, proporcionando una visión clara de sus características, capacidades y aplicaciones actuales.

Característica	IA Débil	IA Fuerte
Definición	Sistemas diseñados para realizar tareas específicas sin conciencia o comprensión real.	Sistemas hipotéticos con la capacidad de entender, aprender y aplicar inteligencia de manera generalizada.
Conciencia	No tienen conciencia ni capacidad de entender su existencia.	Poseerían una forma de conciencia y serían capaces de comprender su existencia y contexto.
Capacidad de Aprendizaje	Especializados en tareas concretas, dependientes de datos y algoritmos.	Capacidad de aprendizaje y adaptación generalizados a una amplia gama de problemas y contextos.
Aplicaciones	Asistentes virtuales, sistemas de recomendación, diagnóstico médico.	Teóricos; podrían incluir decisiones autónomas, resolución de problemas complejos, razonamiento abstracto.
Estado Actual	Ampliamente implementados y en uso activo.	Mayormente teóricos y objeto de investigación y debate.

Tabla 1.1: Comparación entre IA Débil y Fuerte

Dentro de la IA débil existen dos categorías, el aprendizaje máquina y el aprendizaje profundo.

El aprendizaje máquina o Machine Learning (ML) implica el aprendizaje de patrones ocultos dentro de los datos (minería de datos) y su posterior uso para clasificar o predecir un evento relacionado con el problema indica (Alloghani et al., 2020). Se basa en la idea de que los sistemas pueden ser programados y con los datos aprender a identificar patrones y tomar decisiones. Algunos ejemplos de ML son:

- Random Forest
- Regresión lineal y logística
- Clustering

- Máquinas de Soporte Vectorial (SVM)

La segunda categoría corresponde al aprendizaje profundo, esta categoría se centra en el uso de redes neuronales con varias capas ocultas, conocidas como redes profundas. Algunos ejemplos de aprendizaje profundo son:

- Redes neuronales convolucionales
- Redes neuronales recurrentes
- Redes generativas adversarias
- Transformers

En nuestro caso, adoptamos el enfoque de la IA débil, en específico el aprendizaje automático y el aprendizaje profundo, ya que utilizamos grandes cantidades de datos para que una arquitectura de red neuronal aprenda sobre cómo procesar y utilizar esta información para resolver la tarea de generar cavidades F-P a partir de un set de datos de entrada.

Actualmente hay un debate en si se podrá tener alguna vez un programa capaz de considerarse como una IA fuerte, algunos autores como (Chalmers, 1996) que en su trabajo aborda la complejidad del problema de la conciencia que es un elemento fundamental para la IA fuerte argumenta que obtener un programa de estas características si es posible ya que si la dinámica cognitiva es computable, entonces el tipo correcto de organización computacional dará lugar a la conciencia, sin embargo hasta el día de hoy no hay reportes sobre que aquello se haya logrado.

Siguiendo la idea anterior, (Marcus, 2018) realiza una critica hacia el estado actual de la inteligencia artificial sobre lo poco o nada que se ha avanzado en una IA fuerte y lo mucho que se ha avanzado en IA's débiles, lo que lleva a un conclusión que, si bien la IA (en particular el aprendizaje profundo) es una poderosa herramienta, actualmente no es más que otra herramienta estadística y que por su naturaleza estadística presenta problemas, por ejemplo, que se necesite una gran cantidad de datos para que una tarea, dentro de un rango definido sea completada con éxito, pero la capacidad de interpolar y extrapolar dista mucho de lo que hace un ser humano, de hecho, argumenta que el ser humano con muy pocas muestras puede realizar una tarea que una IA débil necesitaría mucha información.

Por lo tanto el concepto IA se compone de categorías y subcategorías 1.3 cada una aportando técnicas y métodos distintos para resolver una problemática.

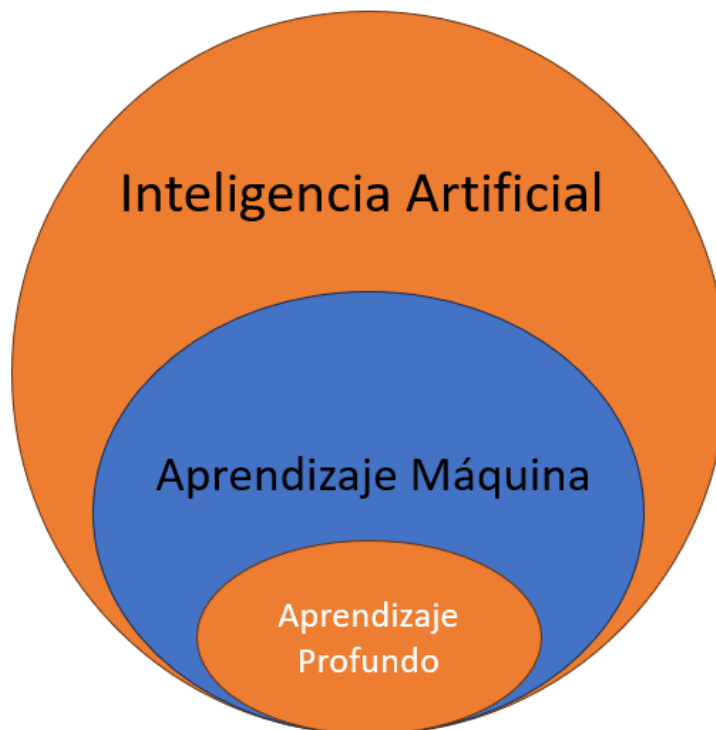


Figura 1.3: Esquema de inteligencia artificial, aprendizaje automático y profundo. Elaboración propia

1.2.4. Aprendizaje supervisado y no supervisado

Antes de detallar qué son las redes neuronales, es crucial determinar primero el esquema de aprendizaje que emplearemos, ya sea supervisado o no supervisado. Basándonos en esta distinción, los problemas relacionados con la inteligencia artificial se pueden clasificar en dos categorías principales, aunque existen otras.

El aprendizaje supervisado es un paradigma en el campo del aprendizaje máquina donde un modelo de IA aprende a partir de un conjunto de datos etiquetados. En este contexto, “datos etiquetados” se refieren a aquellos que vienen acompañados de la salida deseada. Es decir, cada elemento del conjunto de entrenamiento incluye no solo la información de entrada, sino que también la salida correcta asociada a esa entrada. Esta configuración permite que el modelo aprenda la relación entre las entradas y sus salidas correspondientes, a partir de estos ejemplos es que se construye el conjunto de datos de aprendizaje del modelo.

En la práctica, en un problema de regresión en donde el objetivo principal es aprender la distribución D de los datos del conjunto de entrenamiento se busca obtener una función de mapeo f que, dada una entrada x , prediga la salida correspondiente y . Con lo anterior

podemos definir lo siguiente.

Un vector de n entradas X

$$X = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \quad (1.2)$$

y un conjunto de n salidas Y

$$Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \quad (1.3)$$

En donde para cada x , existe (en general) un y .

por lo tanto podemos definir f como

$$f(x_i) = y_i \quad (1.4)$$

Con i la i -ésima muestra.

Finalmente, en general el problema de regresión se puede ver como

$$S = \{(x_i, y_i), \dots (x_n, y_n)\} \quad (1.5)$$

Obteniendo la distribución S que nos sirve para mapear la distribución D original

En problemas donde la función f es costoso de obtener con métodos numéricos, los problemas de regresión en un espacio acotado se pueden solucionar con aprendizaje supervisado.

Por otra parte, el aprendizaje no supervisado se refiere a un conjunto de algoritmos de aprendizaje máquina que trabajan sobre datos sin etiquetas previas. Su principal característica es la ausencia de una variable objetivo o etiquetas que guíen el proceso de aprendizaje. Por lo tanto, el aprendizaje no supervisado se centra en la detección de patrones y la estructura inherente en los datos, dentro de este grupo existen técnicas como el clustering y la reducción de dimensionalidad.

El concepto de *clustering* se refiere a la técnica de segmentación de datos, con el propósito de agrupar un conjunto de datos en diferentes categorías o clústeres. Esto se hace de tal manera que los datos dentro de cada grupo presentan una mayor similitud entre sí en comparación con los de otros grupos. Esta técnica es una forma de clasificación automática que resulta especialmente valiosa cuando se trabaja con datos no etiquetados (Hastie et al., 2009). Una de las técnicas de *clustering* más conocidas y ampliamente utilizadas es el algoritmo K-Means, que tiene como objetivo dividir el conjunto de datos en K grupos distintos, minimizando la suma de las distancias al cuadrado entre los puntos

de datos y el centroide de su clúster asignado. El centroide del clúster se define como la media de los puntos que pertenecen a ese clúster. La función objetivo del algoritmo K-Means, a menudo referida como la inercia, se define como:

$$J = \sum_{i=1}^k \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2, \quad (1.6)$$

donde $\boldsymbol{\mu}_i$ representa el centroide del grupo S_i . El proceso iterativo del K-Means continúa ajustando los centroides y reasignando los puntos de datos a los clústeres más cercanos, hasta que la asignación de clústeres ya no cambia significativamente o se alcanza un número predeterminado de iteraciones.

Esta entre varias otras como Principal Component Analysis corresponden a técnicas de aprendizaje no supervisado pero su detalle está fuera del alcance de esta tesis. además podemos agrupar y distinguir los tipos de aprendizaje con la figura 1.4

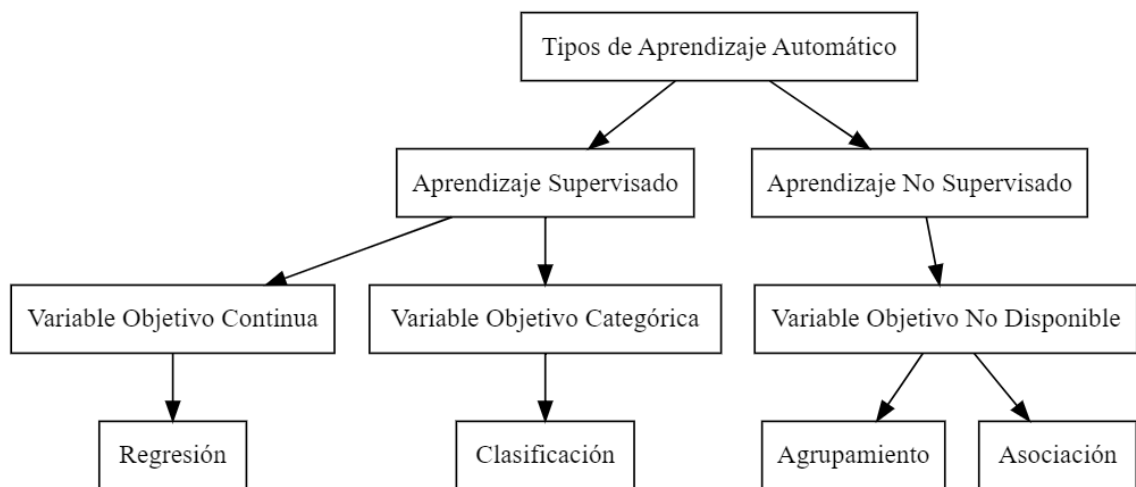


Figura 1.4: Tipos de aprendizaje en ML. Elaboración propia

1.2.5. Redes neuronales

Las redes neuronales tienen como origen la idea de simular el comportamiento del cerebro humano en el proceso de aprendizaje, donde las neuronas biológicas tienen la capacidad de aprender en conjunto para resolver problemas. En (Bishop, 1994) se muestra el primer acercamiento matemático hacia las redes neuronales mediante el modelo de McCulloch y Pitts que más tarde se llamaría perceptrón o neurona. Una neurona (artificial) es la unidad básica de una red neuronal.

Una neurona artificial se modela como una función no lineal que transforma un conjunto de variables de entrada x_i ($i = 1, 2, \dots, d$), siendo d el número de entradas para dicha neurona. Los componentes principales de una neurona son:

- Un conjunto de variables de entrada X , con una dimensión d .
- Un conjunto de pesos W , también de dimensión d , que modula la importancia de cada entrada.
- Una función de activación f , que determina si la neurona se activa o no en función del valor resultante de combinar las entradas y los pesos.
- Un término de sesgo o *bias*, un valor constante añadido al sumatorio de entradas y pesos, que permite ajustar la salida de la neurona incluso cuando todas las entradas son cero.

La operación de una neurona se puede visualizar claramente en la Figura 1.5, y matemáticamente se representa como:

$$y = f \left(\sum_{i=1}^d x_i \cdot w_i + b \right), \quad (1.7)$$

donde y es la salida de la neurona, x_i son las entradas, w_i son los pesos asociados a estas entradas, b es el sesgo, y f es la función de activación que decide el estado de activación de la neurona. Esta formulación captura la esencia del perceptrón, demostrando cómo las redes neuronales derivan su poder computacional a través de la simple, pero poderosa idea de combinar linealmente entradas ponderadas y aplicar una transformación no lineal.

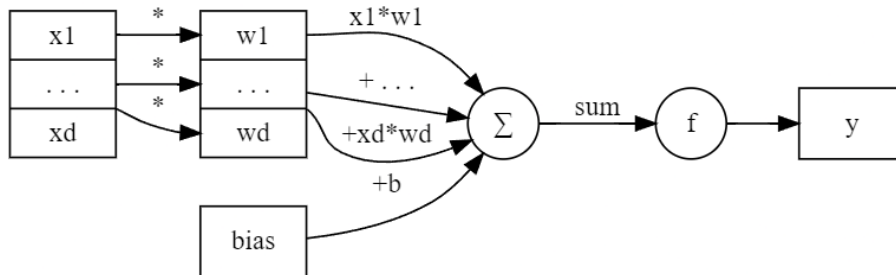


Figura 1.5: Esquema de un Perceptrón. Elaboración propia

Las funciones de activación f , tales como Sigmoide, Tangente Hiperbólica (Tanh), Rectified Linear Unit (ReLU), su variante Leaky ReLU, Softmax, y la función lineal, desempeñan un papel crucial al introducir no linealidades en el modelo, permitiendo que la red neuronal aprenda patrones complejos y realice tareas como clasificación y regresión

con alta eficiencia. A continuación, se describen brevemente estas funciones junto con sus expresiones matemáticas:

- **Sigmoide**

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (1.8)$$

Transforma los valores de entrada a un rango entre 0 y 1, útil para la clasificación binaria.

- **Tangente Hiperbólica (Tanh)**

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (1.9)$$

Similar a la sigmoide, pero mapea los valores entre -1 y 1, ofreciendo una salida centrada en cero.

- **Rectified Linear Unit (ReLU)**

$$\text{ReLU}(x) = \text{máx}(0, x) \quad (1.10)$$

Proporciona una salida lineal para valores positivos y cero para valores negativos, mitiga el problema de desvanecimiento de gradiente.

- **Leaky ReLU**

$$\text{Leaky ReLU}(x) = \text{máx}(0, 0.01x, x) \quad (1.11)$$

Una variante de ReLU que permite un pequeño gradiente cuando $x < 0$, ayudando a mantener activas las neuronas que de otro modo estarían inactivas.

- **Softmax**

$$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^K e^{x_j}} \quad (1.12)$$

Convierte un vector de valores en una distribución de probabilidades, ideal para clasificación multiclase.

- **Linear**

$$\text{Linear}(x) = x \quad (1.13)$$

Función de identidad que no altera la entrada, útil en la capa de salida para problemas de regresión.

Las formas de estas funciones de activación se ilustran en la Figura 1.6.

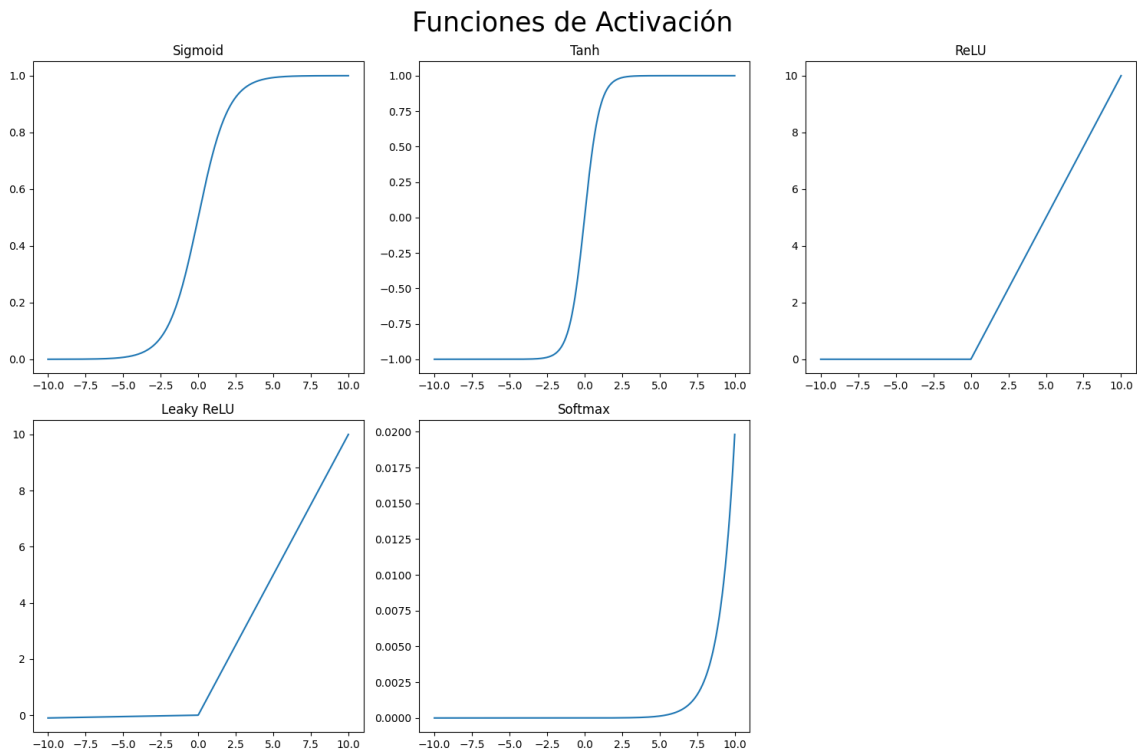


Figura 1.6: Funciones de activación. Elaboración propia

La elección de la función de activación adecuada depende de la naturaleza específica del problema a resolver. La tabla 1.2 proporciona un resumen sobre la utilidad de cada función en contextos de clasificación y regresión:

Función de Activación	Utilidad en Clasificación	Utilidad en Regresión
Sigmoide	Muy útil para clasificación binaria (salidas entre 0 y 1)	Menos común en regresión
Tangente Hiperbólica (Tanh)	Útil para clasificación binaria (salidas entre -1 y 1)	Puede usarse en regresión para salidas normalizadas
Rectified Linear Unit (ReLU)	Ampliamente utilizado en clasificación multiclase y binaria	Común en regresión, especialmente eficaz para evitar el desvanecimiento del gradiente
Leaky ReLU	Variante de ReLU útil en clasificación para evitar neuronas inactivas	Similar a ReLU, útil en regresión
Softmax	Esencial para clasificación multiclase (probabilidad de cada clase)	No se utiliza en regresión
Linear	Raramente se usa en clasificación	Preferida en regresión para la capa de salida

Tabla 1.2: Tabla comparativa de funciones de activación para clasificación y regresión

Habiendo visto ya qué es un perceptrón, una red neuronal es un conjunto o capas de perceptrones trabajando en conjunto para producir un output esperado. Se debe considerar que cada capa de perceptrones puede tener distintas funciones de activación con respecto a otra capa, por ejemplo, una capa puede tener como función de activación ReLU y la siguiente puede tener como función de activación una función lineal. En la figura 1.7 se retrata cómo se ve una red neuronal de múltiples capas.

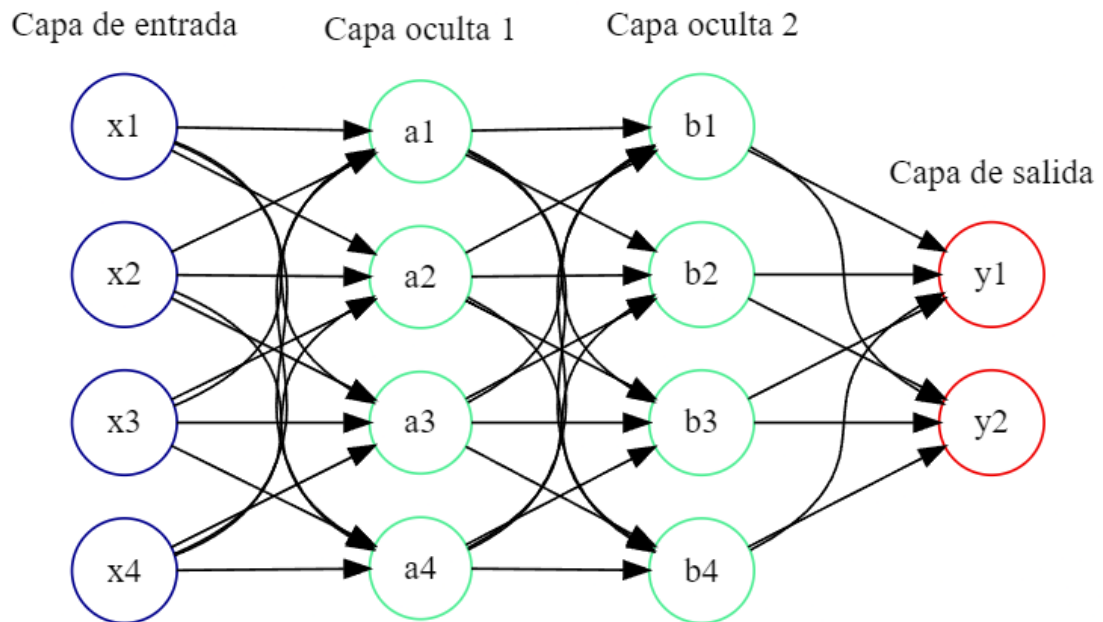


Figura 1.7: Red Neuronal Básica. Elaboración propia

Donde las neuronas x_1 a x_4 actúan como neuronas de entrada, encargadas de recibir los datos iniciales para el procesamiento. Luego, las neuronas desde a_1 hasta a_4 , junto con las neuronas b_1 a b_4 , conforman lo que se denomina capas ocultas. Las capas ocultas son conjuntos de neuronas que procesan los datos recibidos de la capa de entrada, aplicando funciones de activación para extraer características y patrones. Por último, las neuronas y_1 y y_2 constituyen la capa de salida, que tiene la función de entregar los resultados esperados.

Para que una red neuronal aprenda la distribución de los datos de interés es necesario que pase por un proceso conocido como entrenamiento, en el proceso de entrenamiento la red neuronal hará predicciones basadas en los datos de entrada y las compara con los resultados esperados. Para evaluar cuán precisas son estas predicciones y, por ende, el rendimiento de la red, se emplea una métrica denominada función de pérdida o *loss function*, el tipo de función de pérdida varía en función del tipo de problema que se quiere resolver, en general, los problemas con arquitecturas simples de redes neuronales se pueden clasificar en dos grupos:

1. Regresión
2. Clasificación binaria y multiclase

Para problemas de regresión, el error cuadrático medio o *mean square error* (MSE) es una métrica comúnmente utilizada para evaluar el rendimiento del modelo. Dado un conjunto de N muestras, donde y_i representa las etiquetas verdaderas y \hat{y}_i son las predicciones realizadas por el modelo, el MSE se define como:

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (1.14)$$

Esta fórmula cuantifica la diferencia promedio al cuadrado entre los valores reales y las predicciones del modelo. Cuanto menor sea el valor de MSE, más preciso será el modelo en términos de su capacidad para predecir los resultados dentro del conjunto de datos.

En el caso de clasificación binaria una métrica comúnmente utilizada para evaluar el rendimiento del modelo es la entropía cruzada binaria (*Binary Cross-Entropy*, BCE). Esta métrica es útil para predecir si una muestra pertenece a la clase **A** o **B**, produciendo un valor en el rango de 0 a 1. Valores cercanos a 0 indican que la muestra probablemente no pertenezca a la clase, mientras que valores cercanos a 1 sugieren lo contrario. La BCE se define matemáticamente como:

$$BCE = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \quad (1.15)$$

donde N es el número total de muestras, y_i son las etiquetas verdaderas y \hat{y}_i son las predicciones del modelo.

Para problemas de clasificación multiclase, se utiliza una generalización de la BCE denominada entropía cruzada categórica (*Categorical Cross-Entropy*, CCE), que se define como:

$$CCE = -\frac{1}{N} \sum_{i=1}^N \sum_{c=1}^C y_{ic} \log(\hat{y}_{ic}) \quad (1.16)$$

Aquí, C representa el número de clases, y_{ic} son las etiquetas verdaderas codificadas en formato one-hot, y \hat{y}_{ic} son las predicciones del modelo para cada clase.

En el entrenamiento de redes neuronales para diversos tipos de problemas, ya sea regresión, clasificación binaria o clasificación multiclase, el algoritmo de entrenamiento es el mismo, mientras que principalmente cambian el enfoque del problema, la arquitectura de la red y la función de pérdida utilizada. Este algoritmo esencial es conocido como *Propagación hacia atrás*, un método para el aprendizaje supervisado de redes neuronales que facilita el cálculo del gradiente necesario para la actualización de los pesos de la red.

El proceso de propagación hacia atrás consta de dos fases principales:

1. **Propagación hacia adelante:** Donde las entradas son pasadas a través de la red, capa por capa, para generar las predicciones de salida.
2. **Propagación hacia atrás:** Donde los gradientes son calculados a través de la red en dirección inversa, desde la salida hacia la entrada, para actualizar los pesos en función del error calculado entre las predicciones y las verdaderas etiquetas.

Este método fue presentado en 1974 por Paul Werbos en su tesis doctoral (Werbos, 1974) y posteriormente se han realizado desarrollos y aplicaciones más directas y enfocadas del algoritmo, como se detalla en (Werbos, 1990).

Propagación hacia adelante

El proceso de propagación hacia adelante es el primer paso en el ciclo de entrenamiento de una red neuronal, donde las entradas se procesan para generar una predicción. Consideremos una red neuronal con una sola capa oculta para ilustrar este proceso.

Inicialmente, las entradas x son pasadas a través de la red para calcular la salida de la capa oculta h . Esto se calcula como:

$$h = \sigma(W^{(1)}x + b^{(1)}) \quad (1.17)$$

Aquí, $W^{(1)}$ representa los pesos asociados a las conexiones entre la capa de entrada y la capa oculta, y $b^{(1)}$ es el sesgo de la capa oculta. La función σ es una función de activación.

Una vez calculada la salida de la capa oculta h , el siguiente paso es determinar la predicción final de la red \hat{y} , utilizando los valores de h como entradas para la capa de salida:

$$\hat{y} = \sigma(W^{(2)}h + b^{(2)}) \quad (1.18)$$

En esta fase, $W^{(2)}$ y $b^{(2)}$ son los pesos y el sesgo de la capa de salida, respectivamente.

Propagación hacia atrás

Este paso implica el cálculo de la derivada parcial del error E con respecto a cada peso en la red, seguido de una actualización de los pesos para minimizar el error. Utilizando la ecuación para la salida de la red, calculamos para los pesos de la capa de salida:

$$\frac{\partial E}{\partial W^{(2)}} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial h} \cdot \frac{\partial h}{\partial W^{(2)}} \quad (1.19)$$

Un procedimiento similar se aplica para actualizar los pesos de la capa oculta. Una vez calculadas las derivadas parciales, los pesos se actualizan empleando la regla de descenso del gradiente:

$$W^{(i)} = W^{(i)} - \alpha \frac{\partial E}{\partial W^{(i)}} \quad (1.20)$$

donde α es la tasa de aprendizaje. Este enfoque permite la actualización eficiente de los pesos de la red neuronal.

El algoritmo para ejecutar el proceso descrito se muestra a continuación:

```

Datos: Conjunto de datos de entrenamiento
Resultado: Pesos ajustados de la red
inicialización de pesos y parámetros de la red;
mientras no se cumpla el criterio de parada hacer
  para cada ejemplo de entrenamiento hacer
    Realizar propagación hacia adelante para calcular la salida actual de la red;
    Calcular el error en la capa de salida;
    Realizar propagación hacia atrás para actualizar los pesos:
      ■ Calcular el gradiente del error respecto a cada peso;
      ■ Actualizar los pesos usando el gradiente y la tasa de aprendizaje;
  fin
  Evaluar el rendimiento de la red (opcional);
fin
  
```

Algoritmo 1: Algoritmo de propagación hacia atrás

1.3. Optimizador

En las redes neuronales, el optimizador tiene directa relación con el algoritmo de propagación hacia atrás, ya que define cómo se ajustan los parámetros del modelo (pesos y sesgos) para minimizar la función de pérdida. Los optimizadores se basan en el método de descenso de gradiente, un algoritmo de optimización utilizado para minimizar una función objetivo. La regla de actualización de los pesos mediante el descenso de gradiente se expresa como:

$$\theta = \theta - \eta \nabla_{\theta} J(\theta) \quad (1.21)$$

donde θ representa los parámetros del modelo, η es la tasa de aprendizaje, y $\nabla_{\theta} J(\theta)$ es el gradiente de la función de pérdida J con respecto a los parámetros.

El algoritmo de propagación hacia atrás calcula el gradiente $\nabla_{\theta} J(\theta)$, que indica la dirección de mayor incremento de la función de pérdida. Por lo tanto, para minimizar la pérdida, es necesario mover los parámetros en la dirección opuesta al gradiente.

Existen varias variantes del método de descenso de gradiente, por ejemplo:

- **Descenso de Gradiente Estocástico (SGD):** Este método actualiza los parámetros para cada muestra de entrenamiento usando la siguiente regla:

$$\theta = \theta - \eta \nabla_{\theta} J(\theta; x^{(i)}, y^{(i)}) \quad (1.22)$$

donde $(x^{(i)}, y^{(i)})$ corresponde a la i -ésima muestra de entrenamiento.

- **Descenso de Gradiente con Momento:** Introduce un término de momento para acelerar el SGD en la dirección correcta, utilizando:

$$v_t = \gamma v_{t-1} + \eta \nabla_{\theta} J(\theta) \quad \theta = \theta - v_t \quad (1.23)$$

donde v_t es la velocidad en el tiempo t , y γ es el factor de momento.

Para esta investigación, se utilizará el optimizador *Adam* (Adaptive Moment Estimation), que combina las ventajas del SGD con momento y la adaptación de la tasa de aprendizaje (Kingma & Ba, 2014). Adam ajusta los parámetros del modelo según las siguientes actualizaciones:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \nabla_{\theta} J(\theta) \quad (1.24)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) (\nabla_{\theta} J(\theta))^2 \quad (1.25)$$

donde m_t y v_t son estimaciones de los primeros y segundos momentos del gradiente, respectivamente. Adam incluye correcciones de sesgo para estos momentos:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad (1.26)$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (1.27)$$

Y los parámetros se actualizan como:

$$\theta = \theta - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \quad (1.28)$$

1.4. Técnicas de Regularización

Las técnicas de regularización ayudan a prevenir el sobreajuste, mejorar la generalización del modelo a datos no vistos y aumentar el rendimiento general del modelo. El sobreajuste ocurre cuando un modelo aprende los datos de entrenamiento demasiado bien disminuyendo su capacidad de generalización.

Batch Normalization (BatchNorm) es una técnica que mejora la velocidad, rendimiento y estabilidad del entrenamiento de redes neuronales artificiales al normalizar las activaciones de las capas intermedias (**ioffe2015batch**). Se lleva a cabo normalizando las entradas de cada cap, esto permite que cada capa de la red aprenda de manera más independiente de las otras, facilitando el uso de tasas de aprendizaje más altas y reduciendo la dependencia del proceso de inicialización. La operación de Batch Normalization para una entrada x se define como:

$$y = \text{BN}_{\gamma, \beta}(x) = \gamma \left(\frac{x - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \right) + \beta \quad (1.29)$$

donde x es la entrada a la capa de normalización, μ_B es la media por lote (batch mean), σ_B^2 es la varianza por lote (batch variance), ϵ es una constante pequeña que proporciona estabilidad numérica, y γ, β son parámetros que representan el factor de escala y el desplazamiento.

El **Dropout** es una técnica de regularización propuesta por Srivastava et al. (2014), donde, durante el proceso de entrenamiento, se "apagan" o se "desactivan" neuronas de manera aleatoria (**srivastava2014dropout**). Se define una tasa de Dropout p , usualmente fijada a 0.5, que representa la probabilidad de que cualquier neurona sea eliminada en una pasada hacia adelante durante el entrenamiento. Este proceso se puede describir formalmente como:

$$r_j^{(l)} \sim \text{Bernoulli}(p) \quad (1.30)$$

$$\hat{y}^{(l)} = r^{(l)} * y^{(l)} \quad (1.31)$$

donde $r_j^{(l)}$ es una variable aleatoria independiente para cada neurona j en la capa l , que se toma de una distribución de Bernoulli con probabilidad p de ser 1 (mantener la neurona activa). La operación $*$ representa la multiplicación de elementos. El vector resultante $\hat{y}^{(l)}$ es lo que pasa a la siguiente capa durante el entrenamiento. Durante la fase de prueba o evaluación, no se utiliza el Dropout.

1.5. Descripción de la problemática

A lo largo de este marco teórico, se han explicado términos como diseño inverso, cavidad F-P e inteligencia artificial. La razón de ello es mostrar las bases y el contexto en el que se justifica la problemática a abordar en esta tesis.

El esquema del problema es el siguiente; En la figura 1.8 vemos una cavidad F-P delimitada por los espejos de Plata (Ag) y en su interior encontramos dos materiales distintos, que es la silica (SiO_2) y el TDBC (componente orgánico). A partir de una cavidad F-P se puede obtener el espectro de Transmisión (T), Reflexión (R) y Absorción (A), en este caso mediante el método de matrices de transferencia, cada uno de ellos se refiere a lo siguiente:

- **Transmisión:** Es el fenómeno por el cual la luz pasa a través de la cavidad
- **Reflexión:** Este fenómeno ocurre cuando parte de la luz es reflejada en la cara incidente
- **Absorción:** Es el proceso por el cual la luz es atenuada mientras se propaga en la estructura por ser absorbida en los materiales que la componen.

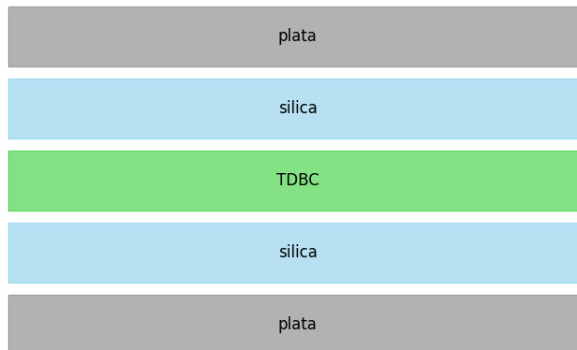


Figura 1.8: Cavidad F-P utilizada. Elaboración propia

En la figura 1.9 vemos el espectro resultante obtenido a partir del método de matrices de transferencia, en el eje X se encuentra la longitud de onda en nanómetros de la de luz que atraviesa la cavidad F-P, en el eje y la intensidad, por conservación de energía, cada trío de puntos debe cumplir la condición de $R + T + A = 1$. Para este caso los espesores de cada una de las subdivisiones de la cavidad vistas de arriba hacia abajo son las siguientes

- 15 nm para Plata (primera capa)
- 31 nm para Silica (segunda capa)
- 23 nm para TDBC (tercera capa)

- 31 nm para Silica (cuarta capa)
- 13 nm para Plata (quinta capa)

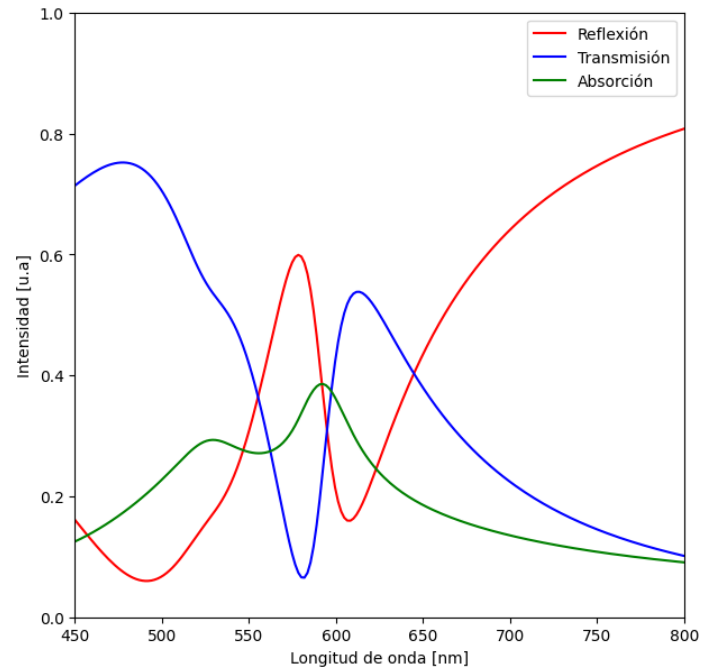


Figura 1.9: Espectro TRA para una cavidad F-P. Elaboración propia

El método de matrices de transferencia funciona correctamente cuando se ve el problema en dirección **cavidad** → **espectro RTA**, sin embargo, encontrar una cavidad con un espectro RTA en específico requiere de una cantidad significativa de prueba y error al variar la estructura de la cavidad. Por lo tanto se requiere generar un método alternativo al de matrices de transferencia tal que se pueda invertir el problema, es decir **espectro RTA** → **cavidad**.

Por lo anterior, en esta tesis se propone utilizar redes neuronales para la resolución del problema debido a que las redes neuronales pueden aprender las relaciones no lineales de la física detrás de la cavidad F-P.

Así, esta tesis propone utilizar redes neuronales para resolver el problema, aprovechando la capacidad de estas de aprender las relaciones no lineales que rigen la física detrás de la cavidad F-P. Nos centraremos en que nuestro modelo de red neuronal pueda predecir los espesores de las capas a partir de un espectro de transmisión T (la línea azul de la figura), manteniendo fijo el orden y el tipo de material de las capas por simplicidad.

1.6. Hipótesis y postulados

Como se mencionó anteriormente el objetivo de esta tesis es desarrollar una herramienta que permita el diseño inverso de cavidades F-P. La siguiente lista de hipótesis y postulados se derivan de este objetivo principal.

Hipótesis: Una red neuronal puede resolver el problema de diseño inverso de cavidades F-P.

1. **Postulado 1:** Las redes neuronales son capaces de aprender la relación funcional entre las propiedades físicas de las cavidades F-P y su comportamiento óptico resultante.
2. **Postulado 2:** Los parámetros de diseño de una cavidad F-P, como la distancia entre espejos y el índice de refracción de los materiales, son predictores significativos de su respuesta óptica.
3. **Postulado 3:** Los conjuntos de datos de diseño de cavidades F-P poseen suficiente diversidad y complejidad para entrenar efectivamente modelos de redes neuronales.

Capítulo 2

Metodología

La validación de la hipótesis planteada requiere un marco metodológico que facilite una comparación con los métodos deterministas tradicionales.

Este marco se desglosará en varias etapas de la investigación:

1. **Recopilación de Datos:** Se detallará cómo se construyen los datos para entrenar la red neuronal.
2. **Construcción del Modelo:** Se describirá la arquitectura de la red neuronal, especificando cantidad de capas, neuronas, las funciones de activación y parámetros del optimizador.
3. **Entrenamiento y validación del Modelo:** Se explicarán los enfoques de validación cruzada y los conjuntos de datos de prueba independientes que se utilizarán para evaluar el rendimiento del modelo.
4. **Análisis de Resultados:** Se detallarán los métodos estadísticos y las métricas de desempeño que se emplearán para interpretar los resultados del modelo.
5. **Discusión:** Se anticiparán los tipos de discusión que se esperan sobre los resultados, incluyendo limitaciones, implicaciones prácticas y posibles direcciones futuras.

Este marco metodológico proporciona una estructura integral para abordar la investigación.

2.1. Diseño de la Investigación

La investigación se basa en la comparación directa entre el modelo de red neuronal y el algoritmo determinista basado en matrices de transferencia. Esta comparación permitirá evaluar las fortalezas y limitaciones de cada enfoque en el diseño inverso de cavidades F-P.

Las variables de interés para la investigación son fundamentales para determinar la viabilidad del modelo propuesto. Las que son:

- **Precisión en la predicción:** La exactitud con la que el modelo puede predecir los parámetros de diseño de la cavidad F-P a partir de un espectro deseado.
- **Capacidad de generalización:** La habilidad del modelo para aplicar lo aprendido a nuevos datos no vistos durante el entrenamiento, lo cual es esencial para determinar su aplicabilidad práctica.

2.2. Herramientas y Tecnologías

Python ha sido elegido como el lenguaje de programación debido a su simplicidad sintáctica y la variedad de librerías para manejo de datos e inteligencia artificial. Para este proyecto, se utiliza la versión 3.10.0, y la gestión de paquetes se realiza a través de PIP.

La biblioteca **PyTorch**, en su versión 2.0, es la librería seleccionada para el diseño y entrenamiento de la red neuronal. PyTorch es preferido sobre otras bibliotecas como Keras por su facilidad de uso y su capacidad para permitir una configuración detallada de las redes neuronales. Además, al instalar esta librería configura automáticamente el uso de la GPU para acelerar el proceso de entrenamiento de la red.

El desarrollo del código, el análisis de datos y las tareas de visualización se llevan a cabo utilizando el entorno interactivo IPython Notebook en el editor de código Visual Studio Code.

Las especificaciones del sistema en el que se realizará el procesamiento computacional son las siguientes:

- **Procesador:** Intel i7-8750H a 2.2GHz con 12 núcleos lógicos.
- **Memoria RAM:** 16GB.
- **Tarjeta Gráfica:** Nvidia Geforce GTX 1070 con diseño Max-Q.

2.3. Datos

Los datos a utilizar en las simulaciones posteriores fueron obtenidos mediante el método de matrices de transferencia, se simuló un esquema de cavidad F-P de cinco capas Ver figura 1.8, cada capa tiene un espesor en [nm] y cada capa está separada por un material distinto.

Los materiales utilizados en los experimentos son Plata, Silica y TDBC.

La elección de la plata para las capas externas se basa en su capacidad para reflejar la luz en una amplia gama de longitudes de onda.

La silica es utilizada como material de capa espaciadora en la cavidad. Una de sus ventajas más significativas es su baja pérdida óptica. Específicamente, la silica no presenta un coeficiente de extinción en la mayoría de las longitudes de onda de interés, lo que la hace ideal para permitir la propagación de ondas electromagnéticas con mínima atenuación.

Finalmente el TDBC se integra por sus propiedades de absorción en el espectro visible, esenciales para la funcionalidad de la cavidad.

Por otra parte, una estructura multicapa de la cavidad F-P permite un mayor grado de control sobre las propiedades ópticas del sistema. Por ejemplo, la combinación de capas de diferentes materiales puede diseñarse para permitir la interferencia constructiva/destructiva en longitudes de onda específicas.

Para la generación de datos se utilizó una plantilla base que contiene el nombre y el grosor de una capa específica dentro de la estructura de la cavidad.

```
1 CavityTemplate = [  
2   {'nombre': 'sustrato', 'espesor': 0},  
3   {'nombre': 'plata', 'espesor': 'x1'},  
4   {'nombre': 'silica', 'espesor': 'x2'},  
5   {'nombre': 'TDBC', 'espesor': 'x3'},  
6   {'nombre': 'silica', 'espesor': 'x4'},  
7   {'nombre': 'plata', 'espesor': 'x5'},  
8   {'nombre': 'aire', 'espesor': 0},  
9 ]
```

El procedimiento de generación de datos fue configurado para llevarse a cabo a través de un número total de 30,000 iteraciones. Durante cada iteración, se generó un conjunto de espesores de capa aleatorios utilizando una distribución $\mathcal{U}(10, 75)$ nanómetros, para cada una de las cinco capas en la plantilla. Los valores de aire y sustrato se mantuvieron en 0 debido a que el método de matrices de transferencia los necesita para hacer el cálculo. La razón del rango 10 y 75 nanómetros es que a partir de los 75 nanómetros en adelante, si la plata posee un espesor muy grande, deja de haber transmisión dentro de la cavidad F-P, por lo que no tiene sentido analizar un espectro T plano cercano a 0.

2.3.1. Exploración de datos

Una vez generados los datos con el método anteriormente descrito, se procede a observar la forma de los datos, tal que se pueda asegurar que el conjunto de datos está bien formado tal que se minimice el sesgo de la red neuronal.

En el gráfico 2.1 se comparan los valores entre la primera y segunda plata, así también la primera y segunda silica, esto para observar si los datos están esparcidos a lo largo de los valores requeridos 10 y 75 nanómetros. Como se puede observar, en ambos casos

para plata y sílice los valores están esparcidos de forma homogénea a lo largo del rango esperado.

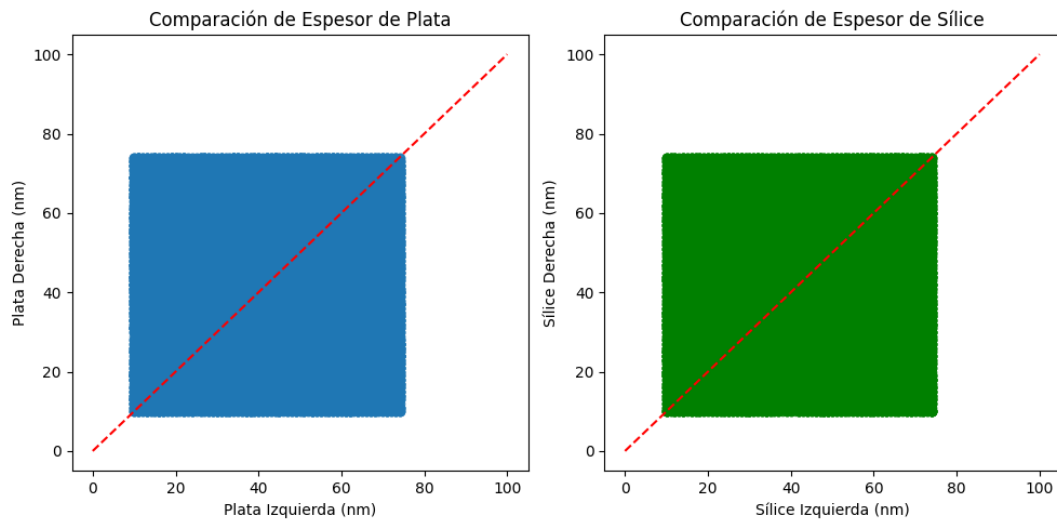


Figura 2.1: Distribución de espesores de plata y sílice en los datos de entrenamiento

Por otra parte, el gráfico 2.2 indica la diferencia absoluta entre espesores de plata y sílice, si observamos el gráfico podemos ver que hay una parte importante del conjunto de datos que su diferencia es menor a 10 nanómetros, esto indica que las cavidades en esos casos tienden a ser simétricas, por otro lado, la cantidad de cavidades que son asimétricas decaen a lo largo del conjunto de datos.

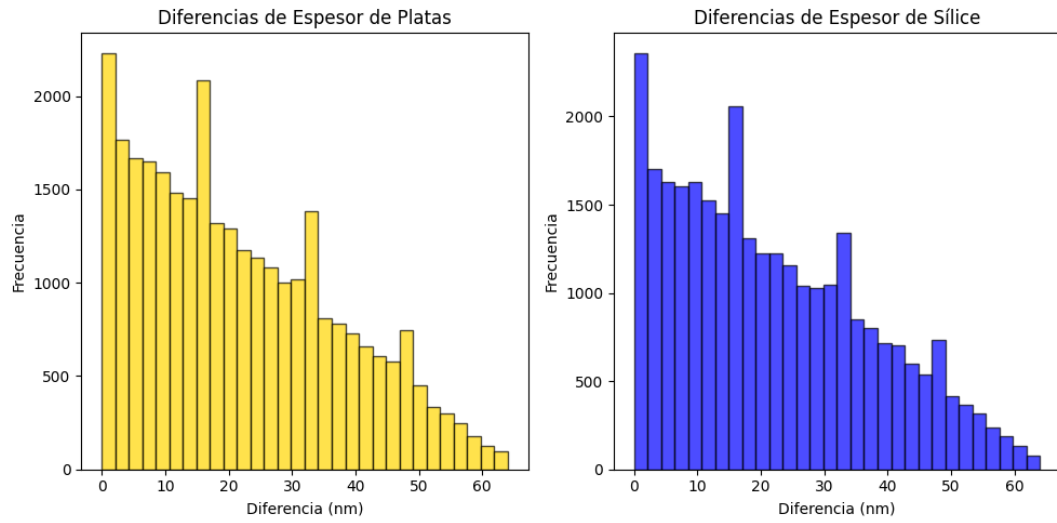


Figura 2.2: Diferencia de espesores de plata y silica

A continuación, la figura 2.3 muestra la distribución de datos del TDBC, se observa lo esperado, una distribución uniforme a lo largo del rango de valores establecido.

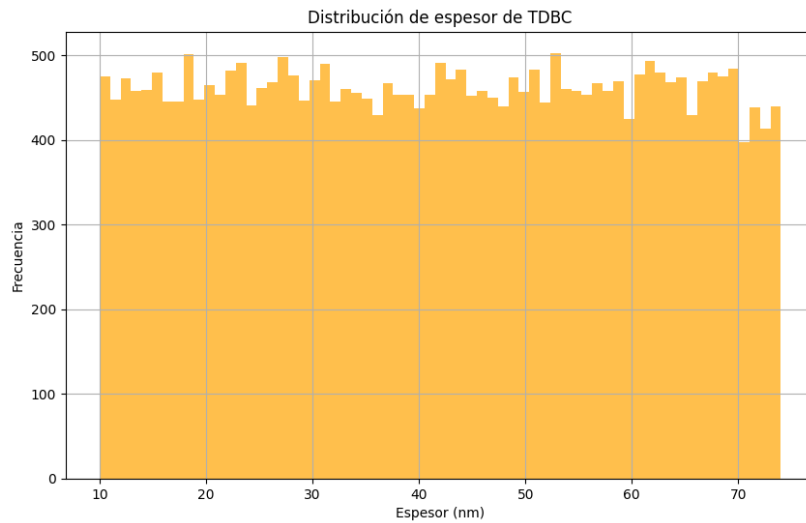


Figura 2.3: Distribución de datos de TDBC

Finalmente, a modo de comprobación de que el conjunto de datos generado sigue la distribución esperada, calculamos la media y varianza teórica de los datos. La media de una distribución uniforme está dada por:

$$E[X] = \frac{a + b}{2} \quad (2.1)$$

con a y b la cota inferior y superior. y la desviación estándar está dada por:

$$\sigma = \sqrt{\frac{(b - a)^2}{12}} \quad (2.2)$$

Reemplazando nos dan los valores teóricos de 45,2 nanómetros para la media y 18,76 nanómetros para la desviación estándar.

Calculando los valores con el conjunto de datos generado nos dan 42,05 nanómetros para la media y 18,77 para la desviación estándar, comprobando así que el conjunto de datos está bien formado.

2.4. Modelo de red neuronal

Anteriormente, se indicó a grandes rasgos la red que se utilizó en esta tesis, desde su funcionamiento, por qué funciona y cómo es su estructura. A continuación, se muestra en detalle cada una de las dos redes neuronales que componen la red Tándem.

Red neuronal Inversa espectro T a espesores (RITE) **Espectro T** → **Espesores**

En el modelo RITE, las capas internas poseen 300 neuronas y funciones de activación como LeakyReLU, además de técnicas de regularización como BatchNorm y Dropout, también en sus capas de salidas está la función de activación ReLU.

La razón por la que la capa de salida es ReLU es porque no tienen sentido físico los valores negativos, por lo tanto, por la forma de ReLU, valores negativos nunca aparecerán como salida válida.

Como método de optimización se seleccionó Adam (Adaptive moment estimation) con los parámetros recomendados en (Ruder, 2016), que son los siguientes;

$$\beta_1 = 0,9$$

$$\beta_2 = 0,999$$

$$\epsilon = 10^{-8}$$

y una tasa de aprendizaje de $lr = 0,001$

2.5. Entrenamiento red neuronal

La partición del conjunto de datos constituye un paso importante en el proceso de entrenamiento de modelos. Entrenar un modelo utilizando la totalidad del conjunto de datos puede llevar a lo que se llama sobre-ajuste. Esto ocurre cuando el modelo realiza

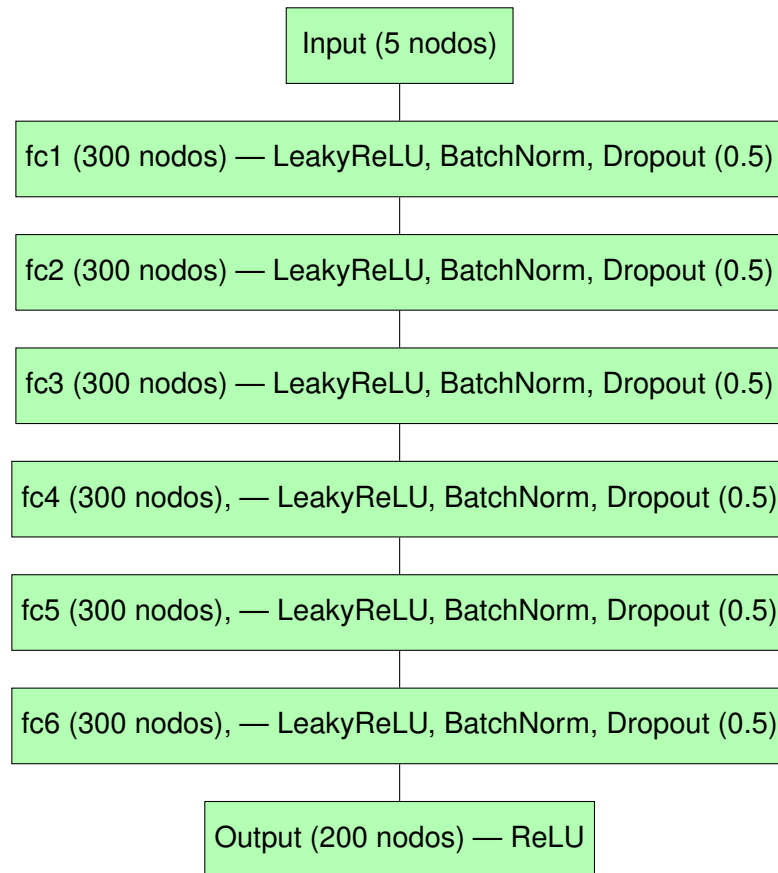


Figura 2.4: Arquitectura de red neuronal

buenas predicciones en el conjunto de entrenamiento pero no en datos que no ha visto, disminuyendo su capacidad de generalizar.

Para evitar esos problemas, una estrategia común implica subdividir el conjunto de datos en tres subconjuntos distintos: entrenamiento, validación y prueba. Cada uno de estos subconjuntos con un propósito diferente:

- **Entrenamiento:** Utilizado para ajustar los parámetros del modelo.
- **Validación:** Empleado para afinar los hiperparámetros del modelo y evitar el sobreajuste.
- **Prueba:** Sirve para evaluar la capacidad de generalización del modelo entrenado sobre datos no vistos.

Además de esta subdivisión, existen técnicas avanzadas que involucran el entrenamiento de múltiples modelos utilizando diferentes particiones de los datos, con el objetivo de evaluar el rendimiento promedio de la red y mejorar su capacidad de generalización. Un método destacado en este contexto es la validación cruzada *K-Fold* (Berrar, 2018).

La validación cruzada *K-Fold* consiste en dividir el conjunto de datos completo en K subconjuntos (o *folds*) de tamaño aproximadamente igual. El proceso de entrenamiento se realiza K veces, utilizando en cada iteración un modelo y *fold* diferente como conjunto de prueba y los restantes $K-1$ *folds* como conjunto de entrenamiento. De esta manera, cada *fold* se utiliza una vez como conjunto de prueba, como también para el entrenamiento y la validación del modelo. Los resultados obtenidos en las K iteraciones se promedian para obtener una estimación del rendimiento del modelo.

En nuestro caso, se utilizó un número de folds $K = 5$.

2.5.1. Entrenamiento red neuronal

Una vez realizado el entrenamiento se grafican las curvas de pérdida i.e el MSE de los K-Folds para entrenamiento y validación.

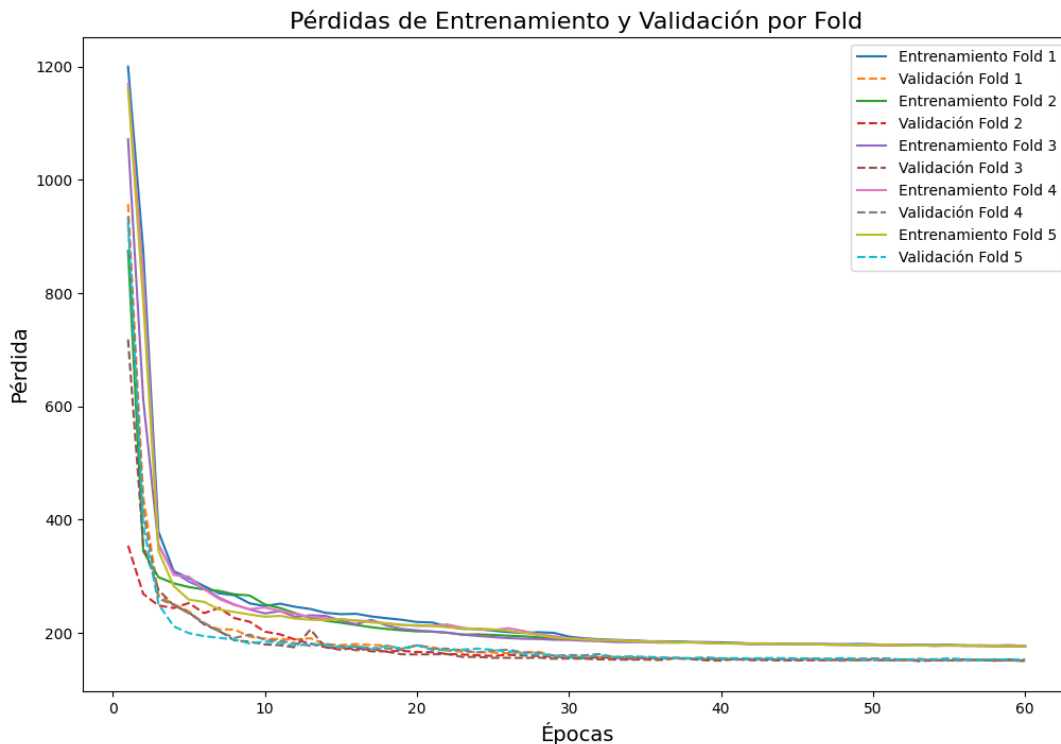


Figura 2.5: MSE para los K-Folds en red RITE

El gráfico presentado en la Figura 2.5 muestra las curvas de pérdidas de entrenamiento y validación utilizando validación cruzada con K-fold. Se puede apreciar que, para cada fold, las pérdidas de entrenamiento y validación disminuyen rápidamente durante las primeras épocas y luego se estabilizan, esto es consistente a través de los cinco folds,

lo que sugiere que el modelo está aprendiendo sin signos de sobreajuste, además, la similitud en las curvas entre los diferentes folds indica que el modelo es estable y tiene un rendimiento consistente independientemente del fold (partición) de los datos utilizada para el entrenamiento y la validación.

Capítulo 3

Resultados

La figura 3.1 presenta los resultados obtenidos luego de entrenar la red neuronal. Los resultados son generados a partir de cavidades del conjunto de datos de prueba, este conjunto no es visto por la red neuronal durante el entrenamiento y la validación.

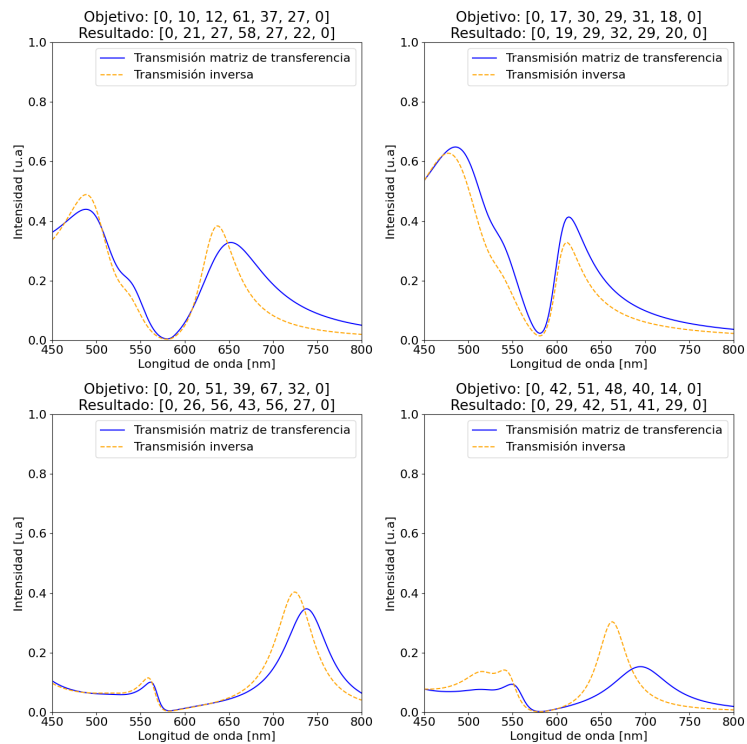


Figura 3.1: Comparación método matriz de transferencia y red neuronal

A la red neuronal se le entregó el espectro de transmisión (color azul) generado a partir de la cavidad F-P con los valores *Objetivo*, luego, la red neuronal a partir del espectro T estimó los espesores dando como resultado los valores *Resultado* y luego se generó el espectro T con la cavidad *Resultado* para luego ser graficado de color amarillo en línea punteada. El proceso descrito se ve reflejado en la figura 3.2

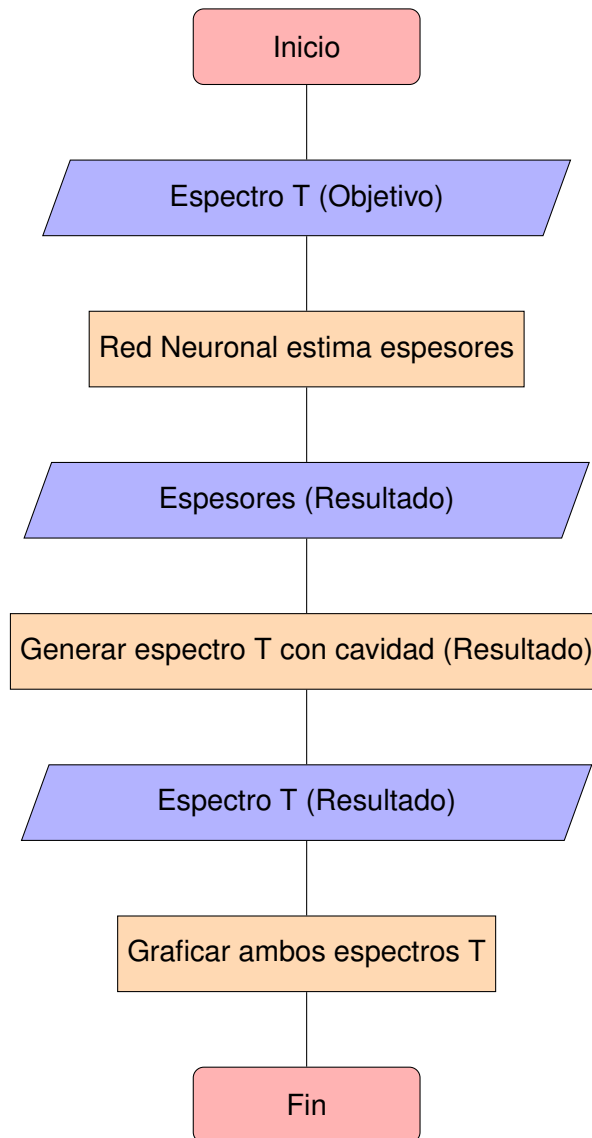


Figura 3.2: Diagrama de flujo de graficar espectros T

Posterior a la obtención de los gráficos anteriores, se procedió a obtener la varianza y el promedio del error cuadrático medio (ECM) y error absoluto medio (EAM) del espectro T objetivo y del espectro T obtenido a partir de la estimación de espesores de la red neuronal

en 10.000 muestras del conjunto de datos de prueba.

Tabla 3.1: Resumen estadístico de los errores de predicción del espectro T

Métrica	Promedio	Varianza
ECM Espectro T	0,0014	$5,6911^{-6}$
EAM Espectro T	0,1599	0,0003

Los resultados obtenidos se resumen en la Tabla 3.1, la cual muestra los valores promedio y la varianza para el ECM y el EAM de la predicción del espectro T por la red neuronal. El valor promedio de ECM de 0,0014 indica que, en promedio, el cuadrado de las diferencias entre los valores predichos por la red y los valores reales es relativamente bajo, lo que indica un buen ajuste del modelo a los datos. Dado que el ECM es más sensible a los errores más grandes debido a su naturaleza cuadrática, un ECM bajo es indicativo de que no existen errores extremadamente grandes en las predicciones. El valor promedio de EAM es de 0,1599 indica que, en promedio, el modelo se equivoca por aproximadamente 0,1599 unidades en la predicción de la intensidad del espectro T, la varianza del ECM es de $5,6911^{-6}$, lo cual muestra que los errores de predicción son consistentes entre diferentes muestras del conjunto de datos, esto es consistente con la baja varianza del ECM.

Los histogramas en la Figura 3.3 muestran la distribución de errores en la estimación de los espesores de las capas de las cavidades de F-P. Estos fueron obtenidos mediante la comparación de 10.000 cavidades de prueba generadas en el rango de entrenamiento y comparadas por capa con las cavidades obtenidas por la red neuronal. El histograma del error de la primera capa de plata muestra una concentración de errores entre 0 y 10 nm, con una cola que se extiende hasta los 50 nm. Lo anterior indica que, aunque el modelo es capaz de predecir con precisión el espesor de la plata, existen situaciones donde el error se incrementa, lo mismo para la sílica. Por otro lado, el histograma del error de TDBC los errores se concentra entre 0 y 10 nm, indicando mayor precisión en las predicciones para este material.

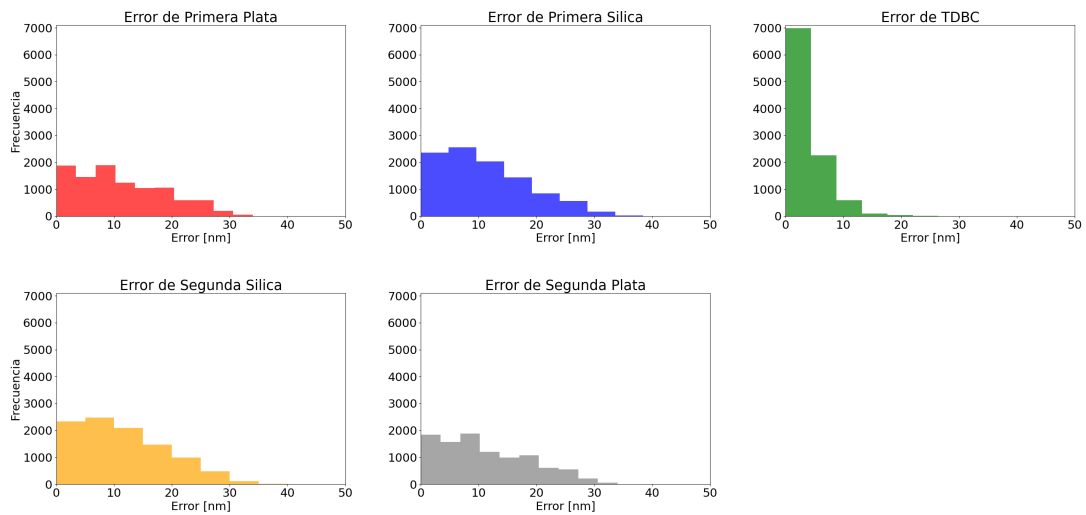


Figura 3.3: Distribución de errores por material en la cavidad F-P. Los histogramas muestran la frecuencia de los errores de predicción.

3.1. Discusión

Observando la función de pérdida de la red neuronal podemos notar que aunque hay una disminución del ECM aproximadamente desde 1200 hasta menos de 200 sigue siendo una pérdida alta, esto llevaría a pensar que la red neuronal no va a predecir valores correctamente, sin embargo al revisar los gráficos de resultados y ver que las predicciones son cercanas pareciera ser incompatible con la pérdida de la red neuronal.

Sin embargo, no se está tomando en consideración la naturaleza del problema, en los problemas de diseño inverso un resultado puede ser obtenido a partir de una (1) o más (n) configuraciones de inicio y nuestro caso de estudio no es ajeno a aquello, ahora, el gráfico de pérdidas de la red neuronal adquiere sentido considerando la naturaleza 1 a n del caso de estudio, en el fondo, la red neuronal está estimando nuevas o configuraciones cercanas a la objetivo que genere la respuesta T esperada, pues, el objetivo final no es estimar con precisión la mejor cavidad, sino que acotar el espacio de búsqueda de todas las cavidades F-P que pueden dar el espectro T objetivo.

Lo anterior se refuerza con los resultados cuantitativos obtenidos del ECM y el EAM, son indicativos de la precisión general del modelo. A pesar de las pérdidas aparentemente altas durante el entrenamiento de la red neuronal, la baja varianza observada en estos errores ($\sigma_{ECM}^2 = 7,6229 \times 10^{-7}$, $\sigma_{EAM}^2 = 9,2370 \times 10^{-5}$) sugiere que el modelo es consistentemente preciso a lo largo de diferentes configuraciones de prueba. También

para reforzar la idea anterior, si analizamos el histograma de errores por capa, en primera instancia es alarmante que el error de las platas y silicas se distribuyan entre 0 y 35 nanómetros, pero considerando lo dicho anteriormente, esto implica que la red neuronal tiene cierta holgura en estas capas para generar cavidades F-P alternativas a la original y que a partir de estas se puedan obtener espectro T similares, algo a destacar de este gráfico es que la holgura en el material de TDBC es más acotada y esto tiene sentido puesto que los gráficos de T en general, tienen picos y valles y estos son causados por el TDBC ya que este es un material absorbente, por lo tanto, a más TDBC es más probable que hayan picos o valles pronunciados, estrechando así la holgura que puede tener este material.

En resumen, aunque las pérdidas de entrenamiento de la red neuronal pueden parecer altas a primera vista, si se analiza profundidad, estas pérdidas no reflejan una incapacidad del modelo para producir soluciones viables. Al contrario, el modelo es capaz de estimar un rango de soluciones que cumplen con los requisitos del espectro T deseado.

Una limitación del modelo presentado es su incapacidad para generalizar el problema a diferentes tipos de materiales. Los experimentos se llevaron a cabo bajo una configuración estática en términos de selección y organización de materiales para las cavidades F-P, lo que restringe el modelo a una relación de correspondencia uno-a-uno entre un espectro T y una configuración particular de cavidad F-P. En este marco, el modelo solo promueve la búsqueda de soluciones con los materiales predeterminados, como la plata en el caso de este estudio. Por lo tanto, si existiera una configuración alternativa que utilizara, por ejemplo, oro en lugar de plata, y esta resultara ser una solución más óptima, el modelo actual no la identificaría, para ello se tendría que cambiar el conjunto de entrenamiento y la arquitectura de la red, añadiendo información de los índices de refracción y respuesta óptica de otros materiales.

3.2. Conclusiones

La investigación ha demostrado que las redes neuronales son efectivas para el diseño inverso en cavidades Fabry-Perot, confirmando su utilidad en predecir configuraciones a partir de respuestas ópticas. El modelo desarrollado ha validado la hipótesis inicial de que la inteligencia artificial puede ser un aporte en el mapeo y predicción de las configuraciones de cavidad F-P a partir de sus respuestas ópticas.

Sin embargo, el modelo presenta limitaciones en la generalización a distintos materiales y configuraciones. Aunque el modelo actual funciona, se requieren más investigaciones para desarrollar un avance en la capacidad de generalización, por lo anterior, se deben explorar arquitecturas que aumenten la flexibilidad del modelo actual, un ejemplo de ello son las redes neuronales Tandem o las redes generativas adversarias (GAN) que pueden resolver el problema $n \rightarrow 1$ con el fin de poder generalizar el problema a varios tipos de material.

En conclusión, esta tesis contribuye al conocimiento y el trabajo interdisciplinario en ciencia de la computación y física de materiales, y es un aporte para futuras investigaciones que podrían llevar a aplicaciones innovadoras en el campo del diseño inverso con IA.

Referencias Bibliográficas

- Abelardo, I., Varela, J., & Urbina, F. H. (2019). *Universidad De Santiago De Chile, Facultad De Ciencia, Departamento de Física, Simulación computacional de la eficiencia óptica de estructuras OLED*.
- Alloghani, M., Al-Jumeily, D., Mustafina, J., Hussain, A., & Aljaaf, A. J. (2020). A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science. En M. W. Berry, A. Mohamed & B. W. Yap (Eds.), *Supervised and Unsupervised Learning for Data Science* (pp. 3-21). Springer International Publishing. https://doi.org/10.1007/978-3-030-22475-2_1
- Ardizzone, L., Kruse, J., Wirkert, S., Rahner, D., Pellegrini, E. W., Klessen, R. S., Maier-Hein, L., Rother, C., & Köthe, U. (2018). Analyzing Inverse Problems with Invertible Neural Networks. <http://arxiv.org/abs/1808.04730>
- Ardizzone, L., Lüth, C., Kruse, J., Rother, C., & Köthe, U. (2019). Guided Image Generation with Conditional Invertible Neural Networks.
- Bäck, T., Breukelaar, R., & Willmes, L. (2005). Inverse design of cellular automata by genetic algorithms: An unconventional programming paradigm. *Unconventional Programming Paradigms*, 3566. https://doi.org/10.1007/11527800_22
- Berrar, D. (2018). Cross-Validation. <https://doi.org/10.1016/B978-0-12-809633-8.20349-X>
- Bishop, C. M. (1994). *Neural networks and their applications*.
- Chalmers, D. J. (1996). *The Conscious Mind: In Search of a Fundamental Theory (2nd edition)*. Oxford University Press.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Bengio, Y., et al. (2014). Generative adversarial networks. *arXiv preprint arXiv:1406.2661*.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning*. Springer Series in Statistics.
- Kim, S., Shin, J. M., Lee, J., Park, C., Lee, S., Park, J., Seo, D., Park, S., Park, C. Y., & Jang, M. S. (2021). Inverse design of organic light-emitting diode structure based on deep

- neural networks. *Nanophotonics*, 10, 4533-4541. <https://doi.org/10.1515/nanoph-2021-0434>
- Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *CoRR*, *abs/1412.6980*. <https://api.semanticscholar.org/CorpusID:6628106>
- Kohavi, R., & Provost, F. (1998). Glossary of terms. *Machine Learning—Special Issue on Applications of Machine Learning and the Knowledge Discovery Process. Machine Learning*, 30, 271-274. <https://doi.org/10.1023/A:1017181826899>
- Liu, Z., Raju, L., Zhu, D., & Cai, W. (2019). A hybrid strategy for the discovery and design of photonic nanostructures. *arXiv:1902.02293 [physics]*.
- Liu, Z., Zhu, D., Rodrigues, S. P., Lee, K. T., & Cai, W. (2018). Generative Model for the Inverse Design of Metasurfaces. *Nano Letters*, 18, 6570-6576. <https://doi.org/10.1021/acs.nanolett.8b03171>
- Marcus, G. (2018). Deep Learning: A Critical Appraisal.
- Molesky, S., Lin, Z., Piggott, A. Y., et al. (2018). Inverse design in nanophotonics. *Nature Photonics*, 12, 659-670. <https://doi.org/10.1038/s41566-018-0246-9>
- Perot, A., & Fabry, C. (1899). On the Application of Interference Phenomena to the Solution of Various Problems of Spectroscopy and Metrology., 9, 87. <https://doi.org/10.1086/140557>
- Peurifoy, J., Shen, Y., Jing, L., Yang, Y., Cano-Renteria, F., DeLacy, B. G., Joannopoulos, J. D., Tegmark, M., & Soljačić, M. (2018). Nanophotonic particle simulation and inverse design using artificial neural networks. *Science Advances*, 4(6). <https://doi.org/10.1126/sciadv.aar4206>
- Ruder, S. (2016). An overview of gradient descent optimization algorithms. *CoRR*, *abs/1609.04747*. <http://arxiv.org/abs/1609.04747>
- Saleh, B. E. A., & Teich, M. C. (2007). *Fundamentals of Photonics* (2.^a ed.). Wiley-Interscience.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56), 1929-1958. <http://jmlr.org/papers/v15/srivastava14a.html>
- Werbos, P. (1974). *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Science. Thesis (Ph. D.). Appl. Math. Harvard University* (Tesis doctoral).
- Werbos, P. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10), 1550-1560. <https://doi.org/10.1109/5.58337>
- Zangeneh-Nejad, F., Sounas, D. L., Alù, A., & Fleury, R. (2020). Analogue computing with metamaterials. *Nature Reviews Materials*, 6(3), 207-225. <https://doi.org/10.1038/s41578-020-00243-2>

Zunger, A. (2018). Inverse design in search of materials with target functionalities. *Nature Reviews Chemistry*, 2(4), 0121. <https://doi.org/10.1038/s41570-018-0121>