



UNIVERSIDAD DE EXTREMADURA

CENTRO UNIVERSITARIO DE MÉRIDA

GRADO EN INGENIERÍA INFORMÁTICA EN  
TECNOLOGÍAS DE LA INFORMACIÓN

TRABAJO FIN DE GRADO

DETECCIÓN DE MICOTOXINAS EN HIGOS  
FRESCOS MEDIANTE TÉCNICAS DE  
INTELIGENCIA ARTIFICIAL

PABLO SETRAKIAN BEARZOTTI

Mérida, Junio de 2024



UNIVERSIDAD DE EXTREMADURA

CENTRO UNIVERSITARIO DE MÉRIDA

GRADO EN INGENIERÍA INFORMÁTICA EN  
TECNOLOGÍAS DE LA INFORMACIÓN

TRABAJO FIN DE GRADO

DETECCIÓN DE MICOTOXINAS EN HIGOS  
FRESCOS MEDIANTE TÉCNICAS DE  
INTELIGENCIA ARTIFICIAL

**Autor:** Pablo Setrakian Bearzotti  
**Fdo:**

**Director:** Francisco Chávez de la O  
**Fdo:**

# Agradecimientos

En primer lugar, quiero expresar mi agradecimiento a mis padres por haberme animado a estudiar Ingeniería Informática y por brindarme su apoyo incondicional a lo largo de toda mi formación académica.

Asimismo, deseo agradecer al profesor Francisco Chávez de la O por sus inspiradoras clases, que despertaron en mí un interés por la inteligencia artificial, así como por haberme propuesto participar en este proyecto.

# Resumen

La seguridad alimentaria representa uno de los desafíos más críticos de la actualidad, especialmente en el contexto de la detección temprana de contaminantes que pueden representar riesgos significativos para la salud humana. Las micotoxinas, sustancias tóxicas producidas por hongos como el *Aspergillus flavus*, constituyen una amenaza considerable en la cadena alimentaria, siendo clasificadas por la Agencia Internacional para la Investigación del Cáncer como sustancias carcinógenas del grupo 1.

La detección tradicional de micotoxinas en productos agrícolas requiere métodos invasivos que implican la destrucción de las muestras, lo que resulta en pérdidas económicas significativas y limitaciones en el control de calidad durante el proceso productivo. En este contexto, el desarrollo de técnicas no invasivas para la detección temprana de contaminación por aflatoxinas se presenta como una necesidad imperante para la industria agroalimentaria.

Este Trabajo Fin de Grado se centra en el desarrollo de un sistema de inteligencia artificial capaz de detectar la contaminación por micotoxinas en higos frescos mediante el análisis de imágenes hiperespectrales y técnicas de inteligencia artificial. El proyecto aborda específicamente la clasificación de estados de enfermedad en frutos de higuera, utilizando un algoritmo genético junto a una red neuronal para identificar la presencia de *Aspergillus flavus* en diferentes concentraciones de contaminación.

La metodología propuesta se estructura en múltiples fases, comenzando con la localización y segmentación de higos individuales mediante detección RGB, seguida de la selección de bandas espectrales más informativas utilizando un algoritmo genético, y culminando con el procesamiento de parches espectrales completos mediante transformadas wavelet. Esta aproximación multimodal permite aprovechar tanto la información espectral específica como las características espaciales de las imágenes hiperespectrales.

Los resultados preliminares demuestran la viabilidad del enfoque propuesto para la detección no invasiva de contaminación por micotoxinas, estableciendo un marco de trabajo que contribuirá significativamente a mejorar los estándares de seguridad alimentaria en la producción de higos, especialmente relevante para regiones productoras como Extremadura, que representa el 55.5 % de la producción nacional española.

***Palabras clave*** - inteligencia artificial, imágenes hiperespectrales, aprendizaje automático, visión por computador, algoritmo genético, micotoxinas, aflatoxinas, seguridad alimentaria, *Aspergillus flavus*, higos, detección no invasiva, transformada wavelet, redes neuronales profundas, redes neuronales convolucionales

# Abstract

Artificial intelligence is no longer just a futuristic promise, but a reality that is redefining the way we interact with the world and conduct business. From healthcare to the manufacturing industry, AI has demonstrated its ability to radically transform diverse fields, unleashing a series of changes that are revolutionising today. Artificial intelligence will enable the automation of a wide range of complex processes that currently still require human intervention.

Artificial intelligence which, through methods, techniques and algorithms, provides computers the ability of identifying patterns in massive data and make predictions (predictive analytics). This learning allows computers to carry out specific tasks autonomously, i.e. without the need to be programmed.

The constant innovation and evolution of the emerging technologies, paired with the use by the vast majority of the world's population, has led to organisations exploiting the power of the automatic learning methods to help them extract better quality information, increase productivity, reduce the costs and get more value from their data.

In an ever-evolving world where technology is redefining the way we live, agriculture is no exception. Artificial intelligence is emerging as a key tool to help farmers adapt to the effects of climate change and lessen its impact. By capitalising on artificial intelligence, farmers can make informed, data-driven decisions and anticipate the climate shifts.

This research deals with sustainable production systems in the cultivation of fig trees and the study of water needs in order to design irrigation strategies using a set of automatic learning techniques belonging to the world of artificial intelligence.

Lastly, once the results generated by the entire system created have been obtained, these will be analysed and the solutions to the problems presented will be identified. These will allow us to observe the degree of efficiency of the experiments developed and to know if they meet the proposed targets.

***Keywords*** - artificial intelligence, hyperspectral images, machine learning, computer vision, genetic algorithm, mycotoxins, aflatoxins, food safety, *Aspergillus flavus*, figs, non-invasive detection, wavelet transform, deep neural networks, convolutional neural networks

# Índice general

Agradecimientos	I
Índice de figuras	VII
Índice de tablas	IX
<b>1. Introducción</b>	<b>1</b>
1.1. Introducción . . . . .	1
1.2. Motivación . . . . .	2
1.3. Objetivo general . . . . .	2
1.4. Objetivos específicos . . . . .	3
1.5. Planificación . . . . .	4
1.6. Organización del documento . . . . .	4
<b>2. Estado del Arte</b>	<b>6</b>
2.1. Introducción . . . . .	6
2.2. HSI en Agricultura de Precisión . . . . .	6
2.3. Imágenes <i>RGB</i> en Agricultura de Precisión . . . . .	8
2.4. ML en Imagenología Agrícola . . . . .	9
<b>3. Desarrollo</b>	<b>12</b>
3.1. Introducción . . . . .	12
3.2. Entorno de Desarrollo . . . . .	12
3.2.1. Gestión de Entornos y Dependencias . . . . .	12
3.2.2. Entorno de Desarrollo Integrado . . . . .	14
3.2.3. Infraestructura Computacional . . . . .	14
3.2.4. Adquisición de Imágenes Hiperespectrales . . . . .	14
3.3. Localización y Segmentación de Figuras . . . . .	16
3.3.1. Objetivo de la Fase . . . . .	16
3.3.2. Herramientas y Tecnologías Empleadas . . . . .	16
3.3.3. Flujo de Procesamiento . . . . .	18



3.3.4.	Resultados . . . . .	21
3.3.5.	Desafíos y Observaciones Técnicas . . . . .	24
3.4.	Selección de Bandas . . . . .	24
3.4.1.	Objetivo de la Fase . . . . .	24
3.4.2.	Herramientas y Tecnologías Empleadas . . . . .	25
3.4.3.	Flujo de Procesamiento . . . . .	26
3.4.4.	Resultados . . . . .	29
3.4.5.	Desafíos y Observaciones Técnicas . . . . .	32
3.5.	Análisis Hiperespectral mediante Transformadas Wavelet . . . .	34
3.5.1.	Objetivo de la Fase . . . . .	34
3.5.2.	Herramientas y Tecnologías Empleadas . . . . .	35
3.5.3.	Flujo de Procesamiento . . . . .	36
3.5.4.	Resultados . . . . .	39
3.5.5.	Desafíos y Observaciones Técnicas . . . . .	40
<b>4.</b>	<b>Resultados</b>	<b>44</b>
<b>5.</b>	<b>Conclusiones y Trabajo Futuro</b>	<b>45</b>
	<b>Referencias</b>	<b>46</b>

# Índice de figuras

3.1.	Ejemplo de detección y segmentación de higos utilizando <i>Grounding DINO</i> y <i>SAM-2</i> . La imagen muestra las cajas delimitadoras generadas por <i>Grounding DINO</i> y las máscaras de segmentación producidas por <i>SAM-2</i> . . . . .	20
3.2.	Ejemplo de la representación en falso color <i>RGB</i> de un higo individual extraído del cubo hiperespectral. Las bandas espectrales utilizadas para la visualización son 220 (Rojo), 113 (Verde) y 78 (Azul). . . . .	21
3.3.	Ejemplo de anotación en formato COCO para la clase 0. . . .	23
3.4.	Curvas de precisión y pérdida del experimento 10 con las bandas seleccionadas 220, 113 y 78. . . . .	30
3.5.	Curvas de evolución de la aptitud del experimento 10 con las bandas seleccionadas 220, 113 y 78. . . . .	31
3.6.	Visualización de la región de 32×32 píxeles extraída automáticamente del higo, subdividida en 64 parches de 4×4 píxeles. . .	36
3.7.	Ejemplo de firma espectral promedio extraída de un parche de 4×4 píxeles. La gráfica muestra la reflectancia en función de las 448 bandas espectrales. . . . .	37
3.8.	Ejemplo de escalograma generado por la Transformada Wavelet Continua ( <i>CWT</i> ) utilizando la <i>wavelet</i> de <i>Morlet</i> . . . . .	42
3.9.	Curvas de entrenamiento del modelo <i>DenseNet-121</i> para clasificación de escalogramas wavelet. Se muestran la evolución de la pérdida y precisión durante 50 épocas de entrenamiento. .	43
3.10.	Comportamiento errático inicial de las métricas de validación, mostrando fluctuaciones extremas en la pérdida de prueba y precisión de prueba durante el entrenamiento sin planificador. .	43

# Índice de tablas

3.1. Resumen de métricas de los experimentos de selección de bandas.	29
3.2. Resultados del experimento 10 con las bandas seleccionadas 220, 113 y 78. . . . .	29
3.3. Reporte de clasificación del experimento 10 con las bandas seleccionadas 220, 113 y 78. . . . .	30
3.4. Matriz de confusión del experimento 10 con las bandas seleccionadas 220, 113 y 78. . . . .	32
3.5. Configuración de la cabeza clasificadora personalizada para <i>DenseNet-121</i> . . . . .	39
3.6. Reporte de clasificación del modelo basado en transformadas wavelet. . . . .	40
3.7. Matriz de confusión del modelo basado en transformadas wavelet. . . . .	40

# Capítulo 1

## Introducción

### 1.1. Introducción

Las enfermedades y plagas en cultivos representan un desafío económico significativo para los sectores agrícola y alimentario a nivel mundial. Entre las amenazas más graves se encuentran las micotoxinas, sustancias producidas naturalmente por ciertos tipos de hongos bajo condiciones particulares de humedad y temperatura [1]. La presencia de micotoxinas en alimentos constituye un problema serio tanto para la salud humana como animal. La Agencia Internacional para la Investigación del Cáncer (IARC) ha clasificado un grupo de aflatoxinas como sustancias carcinogénicas del grupo 1, siendo la vía común de exposición a micotoxinas la ingesta de alimentos contaminados [2].

El hongo *Aspergillus flavus*, que prolifera a temperaturas entre 12°C y 27°C con 85 % de humedad, se multiplica en diversos alimentos incluyendo maíz, cacahuètes, arroz, frutos secos e higos [3]. Aunque su presencia es típica de climas tropicales, también prolifera bajo ciertas condiciones de riego. El ciclo de crecimiento de la aflatoxina es de entre 3 y 5 días [4]. Incluso si los higos van a ser secados, la introducción de higos infectados con aflatoxinas en el proceso puede provocar la contaminación de otros frutos. Por tanto, la detección de aflatoxinas en el producto fresco se considera crucial tanto para el consumo directo como para su procesamiento posterior.

El cultivo de la higuera (*Ficus carica* L.) tiene sus orígenes en la región de Caria en Asia, habiéndose extendido a otras áreas como la región mediterránea, África y América [5]. En 2023 España fue actualmente el octavo mayor productor mundial, representando el 3.05 % de la producción global.

La región de Extremadura, con 12,771 hectáreas cultivadas, representa el mayor productor en España, alcanzando el 55.5 % de la producción nacional [6]. El aumento en la productividad está vinculado a la adopción de técnicas innovadoras como fertilización, poda, tratamiento del suelo e irrigación. Sin embargo, los cambios en la humedad facilitan la propagación de la micotoxina *Aspergillus flavus*, requiriendo investigación adicional para analizar y prevenir que higos infectados entren en la cadena alimentaria humana.

## 1.2. Motivación

La detección tradicional de aflatoxinas se realiza mediante métodos invasivos que requieren la destrucción de la muestra, o mediante inspección visual en etapas avanzadas de contaminación. Estos métodos presentan limitaciones significativas: son lentos, costosos, y no permiten el análisis en tiempo real durante el proceso productivo [7]. Además, los higos frescos son perecederos, tienen una vida útil limitada y son más sensibles al crecimiento microbiano que los higos secos, alterando la calidad del producto y representando un riesgo serio para la salud humana.

El uso de imágenes hiperespectrales (*HSI*) combinado con técnicas de inteligencia artificial, particularmente el aprendizaje profundo, ofrece una alternativa prometedora. La tecnología *HSI* mide la interacción de un amplio espectro de luz con un objeto determinado, adquiriendo cientos de bandas espectrales contiguas para cada píxel en una imagen. Esta capacidad proporciona información detallada sobre el objeto y revela diferencias sutiles en textura y composición química que no son detectables mediante métodos convencionales.

La necesidad de desarrollar métodos no invasivos y precisos para la detección temprana de contaminación por aflatoxinas en higos frescos es crítica para garantizar la seguridad alimentaria, reducir pérdidas económicas en la cadena de producción, y proteger la salud pública.

## 1.3. Objetivo general

Desarrollar un sistema de inteligencia artificial basado en el análisis de imágenes hiperespectrales para la detección temprana de contaminación por micotoxinas en higos frescos, utilizando técnicas de aprendizaje profundo y algoritmos genéticos para optimizar la selección de características espectrales

relevantes. Asimismo, se emplearán métodos de extracción y transformación de información espectral a partir de regiones de interés en las imágenes, con el objetivo de generar representaciones avanzadas que faciliten la evaluación mediante modelos de redes neuronales profundas.

## 1.4. Objetivos específicos

- Implementar un sistema de detección y segmentación automática de higos individuales en imágenes RGB mediante técnicas de visión por computador, generando máscaras y anotaciones para su posterior extracción de datos hiperespectrales.
- Desarrollar e implementar un algoritmo genético para la selección óptima de las tres bandas espectrales más informativas del cubo hiperespectral, reduciendo la dimensionalidad de los datos mientras se mantiene la capacidad discriminativa.
- Diseñar y entrenar modelos de redes neuronales profundas capaces de clasificar el estado de contaminación de los higos basándose en las bandas espectrales seleccionadas, evaluando diferentes arquitecturas y configuraciones.
- Implementar métodos de extracción y transformación de información espectral a partir de regiones de interés en las imágenes hiperespectrales, generando representaciones optimizadas que mejoren la precisión y eficiencia de los modelos de clasificación.

## 1.5. Planificación

El desarrollo del proyecto se estructura en las siguientes fases principales, diseñadas para abordar progresivamente los desafíos técnicos y científicos:

1. **Preparación y adquisición de datos:** Recolección del dataset de imágenes hiperespectrales incluyendo muestras contaminadas con diferentes niveles de micotoxinas y muestras de control no contaminadas, capturadas durante un período de dos semanas para garantizar diversidad y robustez.
2. **Detección y Segmentación de Figuras:** Desarrollo del sistema de detección automática de higos individuales mediante modelos de detección de figuras y segmentación aplicados a versiones *RGB* de las imágenes hiperespectrales, generando máscaras y anotaciones.
3. **Selección de bandas:** Implementación de un algoritmo genético para identificar las tres bandas espectrales más informativas del cubo hiperespectral, construyendo imágenes reducidas para el entrenamiento de redes neuronales. La selección se basará en la capacidad de las bandas para discriminar entre higos contaminados y no contaminados.
4. **Extracción y transformación de información espectral:** Implementación de técnicas avanzadas para extraer y transformar la información espectral de las regiones de interés, generando representaciones en formato imagen para su uso en modelos de redes neuronales profundas.
5. **Comparación de resultados:** Evaluación comparativa de los modelos de redes neuronales entrenados con las bandas seleccionadas mediante el algoritmo genético y las representaciones generadas a partir de la extracción y transformación de información espectral, analizando métricas de rendimiento, eficiencia computacional y consumo energético.

## 1.6. Organización del documento

El presente documento se estructura en los siguientes capítulos para presentar de manera sistemática el desarrollo y resultados del proyecto:

- **Capítulo 2. Marco teórico y estado del arte:** Presenta los fundamentos teóricos de las imágenes hiperespectrales, técnicas de aprendizaje profundo aplicadas a la agricultura de precisión, y una revisión

exhaustiva de trabajos relacionados con la detección de aflatoxinas mediante métodos no invasivos.

- **Capítulo 3. Desarrollo:** Detalla la metodología implementada en cada fase del proyecto, incluyendo la arquitectura del sistema de detección y segmentación, el diseño del algoritmo genético, y la implementación de los modelos de redes neuronales profundas.
- **Capítulo 4. Resultados:** Presenta los resultados experimentales obtenidos en cada fase, incluyendo métricas de rendimiento, análisis comparativo entre diferentes aproximaciones, y evaluación del impacto computacional y energético de los modelos.
- **Capítulo 5. Conclusiones y trabajo futuro:** Resume las contribuciones principales del proyecto, discute las limitaciones encontradas, y propone líneas de investigación futuras para mejorar y extender el sistema desarrollado.



# Capítulo 2

## Estado del Arte

### 2.1. Introducción

La agricultura de precisión ha experimentado una transformación significativa en las últimas décadas, impulsada por el desarrollo de tecnologías emergentes que permiten el monitoreo y análisis automatizado de cultivos [8, 7]. En este contexto, las tecnologías de imagenología avanzada, particularmente *HSI* y *RGB*, han emergido como herramientas clave para la detección temprana de contaminantes y patógenos en productos agrícolas. La integración de estas tecnologías con algoritmos de aprendizaje automático ha abierto nuevas posibilidades para el desarrollo de sistemas de detección no invasivos, precisos y eficientes [7, 9, 10].

Este capítulo presenta una revisión sistemática del estado del arte en tecnologías de detección de contaminantes en productos agrícolas, centrándose en la aplicación de *HSI* y *RGB* en combinación con técnicas de aprendizaje automático. El análisis ofrece el marco teórico necesario para comprender las contribuciones de este proyecto y su relevancia dentro del panorama científico actual.

### 2.2. HSI en Agricultura de Precisión

La imagenología hiperespectral constituye un avance notable en el análisis remoto, ya que permite capturar información extremadamente detallada a través de cientos de bandas espectrales contiguas por cada píxel. Este enfoque mide cómo un amplio rango del espectro electromagnético interactúa con un objeto, proporcionando información sobre su composición química y revelando variaciones sutiles que los métodos convencionales no pueden detectar. La

capacidad de *HSI* para capturar características a través de múltiples bandas permite crear firmas únicas que representan cómo diferentes materiales responden a cada longitud de onda. Según la literatura científica, cuanto mayor es el número de bandas, más detalladas son las características que pueden ser identificadas, aunque no todas las bandas incluyen información relevante para mejorar la precisión de detección [11].

Los sistemas *HSI* típicamente operan en diferentes rangos espectrales, incluyendo el visible (VIS: 400-700 nm), infrarrojo cercano (NIR: 700-1000 nm), infrarrojo de onda corta (SWIR: 1000-2500 nm), y otros rangos especializados. Esta versatilidad espectral permite identificar características específicas de materiales y cambios químicos imperceptibles para el ojo humano o sistemas de imagen convencionales [12].

La aplicación de *HSI* en la detección de contaminantes agrícolas ha mostrado resultados prometedores en numerosos estudios [7, 13]. Estas investigaciones han abordado la identificación de infecciones fúngicas, micotoxinas y otros patógenos que comprometen la calidad y la seguridad de los productos alimentarios. En el contexto específico de detección de aflatoxinas, varios estudios han explorado el potencial de *HSI* para la identificación temprana de contaminación. La investigación ha demostrado que las aflatoxinas, particularmente la Aflatoxina B1 producida por *Aspergillus flavus*, pueden ser detectadas utilizando análisis espectral no invasivo [10]. Estudios recientes han aplicado *HSI* con cámaras *VNIR* (400-1000 nm) y *SWIR* (1000-2500 nm) en diversos cultivos, logrando resultados prometedores en la identificación de muestras contaminadas frente a controles sanos [12].

La distribución superficial de las aflatoxinas representa una ventaja particular para el análisis mediante *HSI*, ya que permite detectar cambios químicos y estructurales en las capas exteriores de los productos agrícolas. Esta característica facilita la implementación de sistemas de detección que no requieren la destrucción de las muestras, manteniendo la integridad del producto para su comercialización.

A pesar de las ventajas evidentes de *HSI*, existen desafíos significativos asociados con su implementación práctica. El principal desafío radica en la alta dimensionalidad de los datos hiperespectrales, que puede representar un obstáculo considerable para los algoritmos de clasificación tradicionales [11, 9]. Los cubos hiperespectrales generan volúmenes masivos de datos complejos que requieren técnicas especializadas de procesamiento y análisis [9]. La gestión de la dimensionalidad espectral requiere estrategias de selección

de características y reducción dimensional para identificar las bandas espectrales más informativas. La eliminación de bandas espectrales redundantes no solo facilita el análisis computacional, sino que también puede mejorar la precisión de clasificación al reducir el ruido y la información irrelevante [14]. Adicionalmente, las condiciones de adquisición de imágenes hiperespectrales requieren un control cuidadoso de factores ambientales como la iluminación, temperatura y humedad, que pueden afectar la calidad y consistencia de los datos espectrales. La calibración y normalización de los datos espectrales son procedimientos críticos para garantizar la reproducibilidad y confiabilidad de los resultados [7].

### 2.3. Imágenes *RGB* en Agricultura de Precisión

En el ámbito agrícola, las imágenes *RGB* siguen siendo la tecnología más extendida [15]. Aunque limitadas en comparación con *HSI*, destacan por su bajo costo, simplicidad de uso y rapidez en el procesamiento, cualidades que han facilitado su implementación a gran escala. Los sistemas *RGB* capturan únicamente tres bandas espectrales correspondientes a los colores primarios, generando representaciones visuales similares a la percepción humana. A pesar de su simplicidad, esta información resulta suficiente para detectar variaciones morfológicas y de color asociadas con infecciones fúngicas u otros contaminantes.

La literatura científica documenta numerosas aplicaciones exitosas de imágenes *RGB* en la detección de contaminantes agrícolas. Estos sistemas han demostrado eficacia particular en la identificación de cambios visuales asociados con infecciones fúngicas, decoloración y alteraciones morfológicas que preceden o acompañan la contaminación por micotoxinas [15]. En el contexto de detección de aflatoxinas, algunos estudios han explorado el uso de imágenes *RGB* para identificar cambios visuales en productos contaminados. Aunque la información espectral limitada de *RGB* puede restringir la detección de cambios químicos sutiles, la tecnología ha mostrado utilidad en la identificación de síntomas visuales avanzados de contaminación fúngica.

Las aplicaciones *RGB* se han extendido también a sistemas de clasificación automatizada para el control de calidad en líneas de producción, donde la velocidad de procesamiento y la simplicidad del sistema son factores críticos. Estos sistemas pueden proporcionar una primera línea de defensa en la detección de productos visiblemente afectados.

Las principales limitaciones de los sistemas *RGB* radican en su capacidad limitada para detectar cambios químicos sutiles que no se manifiestan visualmente. La contaminación por micotoxinas puede ocurrir sin síntomas visuales evidentes en las etapas tempranas, limitando la efectividad de los sistemas *RGB* para la detección precoz. Adicionalmente, los sistemas *RGB* son susceptibles a variaciones en las condiciones de iluminación y pueden requerir normalización cuidadosa para mantener la consistencia en diferentes entornos. La dependencia de características visuales también puede resultar en falsos positivos cuando se presentan variaciones naturales en color o textura que no están relacionadas con contaminación.

## 2.4. ML en Imagenología Agrícola

La aplicación de aprendizaje profundo en el procesamiento de imágenes hiperespectrales ha revolucionado las capacidades de análisis y clasificación en agricultura de precisión [16]. Como subconjunto del aprendizaje automático, el aprendizaje profundo utiliza redes neuronales profundas que consisten en múltiples capas interconectadas de neuronas artificiales capaces de aprender representaciones de alto nivel a partir de datos de entrada.

La implementación de aprendizaje profundo para el procesamiento y análisis de imágenes hiperespectrales fue inicialmente descrita en investigaciones pioneras que propusieron enfoques de clasificación utilizando información espacialmente dominante [17]. Desde entonces, un gran número de estudios han reflejado el interés creciente de la comunidad científica en esta área de investigación.

Las redes neuronales convolucionales tridimensionales han emergido como una arquitectura particularmente efectiva para el procesamiento de imágenes hiperespectrales [14]. Estas redes pueden capturar simultáneamente características espaciales y espectrales, aprovechando la naturaleza tridimensional inherente de los datos hiperespectrales. Las *CNN 3D* operan mediante la aplicación de filtros convolucionales tridimensionales que se desplazan a través de las dimensiones espaciales ( $x$ ,  $y$ ) y espectral ( $z$ ) del cubo hiperespectral. Esta capacidad permite la extracción de características que consideran tanto la variabilidad espacial local como las relaciones espectrales entre bandas adyacentes.

Una alternativa prometedora al uso directo de redes 3D consiste en la aplicación de transformadas matemáticas al espectro antes del procesamien-

to con redes neuronales [10]. Las transformadas *wavelet* han mostrado particular eficacia en este contexto, permitiendo la descomposición del espectro en componentes de frecuencia que pueden ser procesados mediante arquitecturas de red más simples. El enfoque basado en transformadas *wavelet* ofrece ventajas computacionales significativas, ya que permite la conversión de firmas espectrales unidimensionales en representaciones bidimensionales que pueden ser procesadas eficientemente mediante CNN 2D convencionales. Esta metodología puede mantener la información espectral crítica mientras reduce la complejidad computacional del procesamiento.

Las redes neuronales convolucionales bidimensionales representan el estándar establecido para el procesamiento de imágenes *RGB* en aplicaciones agrícolas [15]. Estas arquitecturas han demostrado eficacia excepcional en tareas de clasificación, detección de objetos y segmentación semántica aplicadas a productos agrícolas. Las *CNN 2D* operan mediante la aplicación de filtros convolucionales que capturan características espaciales locales en las imágenes *RGB*. La jerarquía de capas permite la extracción progresiva de características, desde detectores de bordes y texturas en capas tempranas hasta representaciones semánticas complejas en capas profundas.

El desarrollo de arquitecturas especializadas para detección de objetos y segmentación semántica ha facilitado la implementación de sistemas automatizados de análisis agrícola. Arquitecturas como *YOLO (You Only Look Once)* [18], *R-CNN* [19] y sus variantes han demostrado eficacia en la detección y localización automática de productos agrícolas en imágenes *RGB*. Para tareas de segmentación semántica, arquitecturas como *U-Net* [20], *SegNet* [21] y *DeepLab* [22] han mostrado resultados prometedores en la delimitación precisa de regiones de interés en imágenes agrícolas. Estas capacidades son fundamentales para el análisis posterior de características específicas de productos individuales.

La integración de información *RGB* e hiperespectral representa una frontera emergente en el análisis agrícola automatizado [9]. Los enfoques híbridos pueden aprovechar las ventajas complementarias de ambas modalidades: la simplicidad y velocidad del *RGB* para detección y localización, y la riqueza espectral de *HSI* para análisis químico detallado. Estos sistemas multimodales típicamente implementan arquitecturas de procesamiento en cascada, donde la información *RGB* se utiliza para la detección inicial y segmentación de productos, seguida por análisis hiperespectral detallado de las regiones de interés identificadas. Esta estrategia puede optimizar tanto la eficiencia computacional como la precisión de detección.

Las técnicas de fusión de características permiten la combinación sistemática de información extraída de diferentes modalidades de imagen. Estos enfoques pueden implementarse a diferentes niveles de la canalización de procesamiento: fusión temprana (combinación de datos en bruto), fusión intermedia (combinación de características extraídas) o fusión tardía (combinación de decisiones de clasificadores independientes). La fusión efectiva de características *RGB* e hiperespectrales requiere consideración cuidadosa de las diferencias en resolución espacial, rango dinámico y características estadísticas entre las modalidades. Técnicas de normalización y alineamiento espacial son críticas para el éxito de estos enfoques.

El análisis de la literatura revela fortalezas y limitaciones distintas en los enfoques actuales para la detección de contaminantes agrícolas. Los sistemas basados en *HSI* ofrecen capacidades superiores de detección química pero requieren recursos computacionales significativos y equipos especializados costosos. Los sistemas *RGB* proporcionan soluciones más accesibles y eficientes pero con capacidades limitadas de detección temprana.

Las arquitecturas de aprendizaje profundo han demostrado capacidades excepcionales en ambas modalidades, pero su implementación efectiva requiere datasets grandes y representativos que pueden ser costosos tanto en términos de tiempo como de recursos. La transferibilidad de modelos entre diferentes cultivos, condiciones ambientales y sistemas de adquisición permanece como un desafío significativo.

Las oportunidades de innovación identificadas incluyen el desarrollo de arquitecturas híbridas que combinen eficientemente información *RGB* e hiperespectral, la implementación de técnicas de selección inteligente de características espectrales y el desarrollo de sistemas adaptativos que puedan operar efectivamente en condiciones variables de campo [13]. La integración de técnicas de optimización evolutiva, como algoritmos genéticos, para la selección automática de características espectrales representa una dirección prometedora para mejorar tanto la eficiencia como la precisión de los sistemas de detección. Asimismo, la extensión de enfoques basados en transformadas matemáticas desde el análisis de píxeles individuales hacia el procesamiento de regiones espaciales puede capturar información contextual adicional, mejorando la precisión de clasificación mientras se mantiene la eficiencia computacional.

# Capítulo 3

## Desarrollo

### 3.1. Introducción

El desarrollo del proyecto se ha estructurado en múltiples fases secuenciales, cada una diseñada para abordar aspectos específicos del proceso de análisis hiperespectral aplicado a la detección de aflatoxinas en higos frescos. La metodología desarrollada implementa técnicas de visión por computador de última generación combinadas con procesamiento especializado de datos hiperespectrales para crear un sistema automatizado de análisis de muestras.

### 3.2. Entorno de Desarrollo

El proyecto se ha implementado utilizando un entorno de desarrollo adaptado para procesamiento de imágenes hiperespectrales y ejecución de modelos de aprendizaje profundo. A continuación se detallan los aspectos fundamentales de la infraestructura técnica utilizada.

#### 3.2.1. Gestión de Entornos y Dependencias

El proyecto se desarrolló utilizando el lenguaje de programación *Python* [23] en su versión 3.13 con soporte para *Cython* [24], lo que proporcionó ventajas significativas en términos de rendimiento.

Para la gestión de entornos virtuales se utilizó *Conda* [25], un sistema que permite crear espacios de trabajo aislados con versiones específicas de bibliotecas.

Sobre la base de *Conda*, se implementó *UV* [26], un gestor de paquetes

y proyectos para *Python*, extremadamente rápido y escrito en *Rust* [27]. *UV* fue utilizado para la instalación y gestión de paquetes dentro del entorno *Conda*, aprovechando su capacidad para resolver dependencias de manera más eficiente y rápida que las herramientas tradicionales como *pip* o el propio instalador de *Conda*. Esta combinación permitió mantener un entorno consistente y reproducible mientras se optimizaba el tiempo de instalación y actualización de dependencias.

Para cada fase del proyecto se creó un paquete independiente con dependencias específicas según los requisitos de cada etapa. Las principales bibliotecas utilizadas en el proyecto incluyen:

- **PyTorch con torchvision y torchaudio:** Framework principal para implementación de modelos de aprendizaje profundo [28].
- **NumPy y SciPy:** Para operaciones numéricas y manipulación eficiente de matrices [29, 30].
- **Scikit-learn:** Para implementación de algoritmos de aprendizaje automático y métricas [31].
- **Spectral:** Biblioteca especializada para procesamiento de imágenes hiperespectrales [32].
- **OpenCV-Python:** Para operaciones de procesamiento de imágenes y visión por computador [33].
- **Transformers:** Para implementación y uso de modelos basados en arquitecturas de *transformers* [34].
- **Timm:** Colección de modelos preentrenados para tareas de visión por computador [35].
- **Supervision:** Para visualización y análisis de resultados de detección y segmentación [36].
- **PyCocoTools:** Para manipulación de anotaciones en formato *COCO* [37, 38, 39].
- **DEAP:** Para implementación de algoritmos genéticos y evolutivos [40].

Adicionalmente, se incorporaron bibliotecas auxiliares como **addict**, **colorlog**, **gdown**, **split-folders**, **submitit** y **termcolor** para tareas de



gestión de configuración, logging, descarga de modelos pre-entrenados, organización de datos y paralelización de tareas.

La gestión precisa de versiones de estas dependencias resultó crítica para garantizar la compatibilidad entre componentes y estabilidad del entorno de desarrollo.

### 3.2.2. Entorno de Desarrollo Integrado

Para el desarrollo del código se empleó Visual Studio Code (VS Code) como entorno de desarrollo integrado.

### 3.2.3. Infraestructura Computacional

El desarrollo y ejecución del proyecto se realizó en un servidor de alto rendimiento proporcionado por la Universidad de Extremadura, con las siguientes especificaciones técnicas:

- **Procesador:** Intel(R) Xeon(R) Silver 4310 CPU @ 2.10GHz.
- **Memoria RAM:** 512 GB DDR4.
- **Acelerador Gráfico:** 4 × NVIDIA A100 con 40 GB de memoria VRAM cada una, de las cuales se utilizó una para la ejecución de los modelos de aprendizaje profundo.
- **Almacenamiento:** 4 TB en disco SSD NVMe.

### 3.2.4. Adquisición de Imágenes Hiperespectrales

Las imágenes fueron capturadas utilizando una cámara hiperespectral *SPECIM*, específicamente el modelo *FX10 VNIR*, cuyas características técnicas principales incluyen: resolución espacial de 1024 píxeles (800 ancho × 1024 alto), rango espectral de 400 nm a 1000 nm (visible y parte del infrarrojo cercano), 448 bandas espectrales, y un salto espectral de 1.339 nm.

El conjunto de datos comprende 320 higos cosechados de la plantación de la variedad calabacita ubicada en la *Finca La Orden-Valdesequera* (38°51' N, 6°40' W, altitud 184 m) en Guadajira, España, donde *CICYTEX* tiene su sede central. Las imágenes hiperespectrales se capturaron durante un período de 2 semanas, utilizando cada semana 160 higos cosechados en diferentes etapas de madurez.

Cada semana, 160 higos se dividieron en cuatro subconjuntos de aproximadamente 40 especímenes cada uno. El primer grupo correspondió a los controles sanos (Clase 0), mientras que los tres grupos siguientes fueron inoculados con concentraciones de  $10^3$  UFC/mL (Clase 1),  $10^5$  UFC/mL (Clase 2), y  $10^7$  UFC/mL (Clase 3), respectivamente. El proceso de inoculación se realizó mediante inmersión del área durante aproximadamente 3 segundos, siguiendo el protocolo establecido por *CICYTEX*.

Las imágenes hiperespectrales se capturaron después de la inoculación cada 24 horas durante cinco días consecutivos. Entre cada sesión de adquisición, las muestras se almacenaron en una cámara de incubación controlada a 25°C, con humedad relativa entre 80 y 90 % para promover el crecimiento fúngico. Cada clase consistió de 380 imágenes hiperespectrales, generando un total de 1520 imágenes hiperespectrales para el conjunto de datos completo.

Las imágenes hiperespectrales capturadas se almacenan en una estructura de directorios organizada que incluye múltiples archivos asociados a cada adquisición. A continuación, se describe el formato y contenido de los archivos principales:

- **Archivos de datos hiperespectrales (.hdr, .raw):**
  - El archivo `.hdr` contiene metadatos descriptivos de la imagen, como dimensiones espaciales, número de bandas espectrales, rango espectral, y formato de datos.
  - El archivo `.raw` almacena los datos espectrales en bruto, organizados en un formato binario que representa la intensidad de cada banda para cada píxel.
- **Referencias de calibración (DARKREF, WHITEREF):**
  - Los archivos `DARKREF` y `WHITEREF` contienen las referencias oscura y blanca necesarias para la posterior corrección radiométrica de las imágenes hiperespectrales.
- **Imagen .png:**
  - Este archivo representa una visualización en falso color *RGB* generada a partir de tres bandas seleccionadas del cubo hiperespectral.
- **Archivos de metadatos (.xml):**

- Contienen información adicional sobre las condiciones de captura, como fecha, hora, y parámetros experimentales.

Esta estructura permite un manejo eficiente de los datos, facilitando tanto la corrección radiométrica como la integración con el flujo de procesamiento automatizado.

### 3.3. Localización y Segmentación de Figuras

#### 3.3.1. Objetivo de la Fase

La primera fase del proyecto consiste en la creación del conjunto de datos mediante la localización y segmentación automatizada de higos individuales sobre las imágenes creadas a través del falso color *RGB*. El objetivo principal es generar anotaciones precisas en formato COCO [41], que incluyan cuadros delimitadores y máscaras de segmentación para cada higo detectado, y extraer los subcubos hiperspectrales radiométricamente corregidos correspondientes a cada fruto.

Esta fase es fundamental para el flujo de trabajo completo, ya que permite el aislamiento automatizado de las regiones de interés que servirán como imágenes de entrada para el entrenamiento y la inferencia de la *CNN*. La precisión en esta etapa condiciona directamente la calidad de los datos que utilizará la red en las fases posteriores, por lo que resulta esencial garantizar su exactitud.

#### 3.3.2. Herramientas y Tecnologías Empleadas

La implementación de esta fase se basa en la integración de modelos de visión por computador de última generación, complementados con librerías especializadas para el procesamiento de datos hiperspectrales y manipulación de anotaciones.

#### Grounding DINO

*Grounding DINO* [42, 43] es un modelo de inteligencia artificial de última generación especializado en la detección de objetos en imágenes mediante el uso combinado de descripciones textuales e información visual, permitiendo un análisis multimodal avanzado. Gracias a su arquitectura basada en *transformers* [44] y técnicas de aprendizaje profundo, puede localizar y etiquetar

objetos de interés, sin necesidad de entrenamiento específico para cada tipo de objeto, lo que lo hace altamente adaptable para tareas de detección abiertas o *zero-shot* [45].

## SAM2 (Segment Anything Model 2)

*SAM* (*Segment Anything Model*) [46, 47] es un modelo de inteligencia artificial de última generación diseñado para segmentar cualquier objeto en imágenes o videos de manera automática y versátil. Fue desarrollado por *Meta AI* y su objetivo principal es permitir la segmentación de objetos de imágenes y videos sin necesidad de entrenamiento específico para cada clase, usando tecnologías de visión por computador avanzadas y aprendizaje *zero-shot*. Está entrenado en uno de los mayores conjuntos de datos existentes (SA-1B), con 11 millones de imágenes y 1.1 mil millones de máscaras de segmentación, lo que le da una capacidad sobresaliente para generalizar a nuevos contextos visuales.

## COCO (Common Objects in Context)

El formato COCO (Common Objects in Context) [41] es un estándar ampliamente adoptado para el almacenamiento y intercambio de anotaciones en tareas de visión por computador, especialmente en detección de objetos, segmentación de instancias y estimación de poses. Desarrollado por *Microsoft Research*, COCO define una estructura *JSON* [48] que organiza metadatos de imágenes, anotaciones de objetos y categorías de manera eficiente y escalable.

Entre las componentes que definen la estructura del formato COCO, se encuentran las **anotaciones** (*annotations*), las cuales contienen las anotaciones específicas de cada objeto detectado, incluyendo identificadores únicos, referencias a imagen y categoría, coordenadas de bounding box, área, máscaras de segmentación en formato *RLE* (*Run-Length Encoding*), y banderas adicionales como *iscrowd*, que indica si el objeto es parte de un grupo denso.

Para tareas de segmentación de instancias, las máscaras se codifican mediante *RLE*, un algoritmo de compresión sin pérdidas que representa secuencias de píxeles consecutivos como pares (valor, longitud), reduciendo significativamente el espacio de almacenamiento requerido. Las coordenadas de bounding box se especifican en formato `[x, y, width, height]`, donde (x, y) representa la esquina superior izquierda del rectángulo delimitador.

### 3.3.3. Flujo de Procesamiento

La implementación del flujo de trabajo se diseñó siguiendo una arquitectura modular que separa conceptualmente la detección y anotación automatizada de la extracción de subcubos hiperespectrales. Esta separación se materializa en dos módulos principales: el primero responsable de la localización y segmentación de higos individuales sobre las imágenes RGB derivadas, y el segundo encargado de la extracción de los subcubos hiperespectrales correspondientes a cada detección validada.

El sistema adopta un patrón de procesamiento por lotes que opera sistemáticamente sobre la estructura jerárquica del conjunto de datos. Cada directorio de clase (C0, C1, C2, C3) contiene las imágenes hiperespectrales junto con sus archivos de metadatos y referencias de calibración.

#### 1. Detección

El módulo de detección y segmentación constituye el núcleo del flujo automatizado. La implementación utiliza *Grounding DINO* como modelo de detección, empleando la arquitectura con la red principal (*backbone*) *Swin Transformer Base* y el punto de control preentrenado correspondiente. Esta configuración permite al modelo procesar imágenes *RGB* manteniendo su relación de aspecto original mientras utiliza la entrada de texto *fig* (higo en inglés) como descriptor semántico para guiar la detección.

La optimización de los parámetros de inferencia se estableció mediante experimentación empírica, fijando tanto el umbral de confianza de detección como el umbral de similaridad semántica texto-imagen en 0.25. Esta configuración proporciona un equilibrio óptimo entre sensibilidad de detección y precisión para el conjunto de datos específico, minimizando tanto los falsos positivos como los falsos negativos. El proceso de detección implementa una secuencia de validación que comienza con la carga de imágenes seguida de la conversión del espacio de color *BGR* a *RGB* y la extracción automática de metadatos temporales y experimentales del nombre del archivo.

Durante la inferencia, el modelo transforma las imágenes a tensores *PyTorch* aplicando la normalización correspondiente a los parámetros del modelo preentrenado, ejecuta la detección con la entrada de texto especificada y aplica un filtrado geométrico crítico que limita las dimensiones máximas de los cuadros delimitadores a 250×150 píxeles. Esta restricción dimensional resulta fundamental para asegurar la detección de higos individuales y evi-

tar regiones que abarquen múltiples especímenes, un problema recurrente en imágenes con alta densidad de objetos. El post-procesamiento convierte las coordenadas al formato requerido por *SAM-2* y extrae las puntuaciones de confianza asociadas a cada detección.

## 2. Segmentación

La segmentación se realiza mediante *SAM-2*, inicializado con la configuración *Hiera Large* y el punto de control preentrenado correspondiente, utilizando el predictor específicamente diseñado para el procesamiento de imágenes estáticas. El modelo opera sin supervisión de puntos, empleando exclusivamente los cuadros delimitadores generados por *Grounding DINO* como entrada primaria. La configuración para una única máscara por detección asegura la generación coherente, simplificando el procesamiento posterior y manteniendo la consistencia en las anotaciones. La figura 3.1 muestra un ejemplo representativo del proceso de detección y segmentación automatizada implementado.

## 3. Generación de Anotaciones

La generación de anotaciones en formato *COCO* se realizó mediante la construcción sistemática de estructuras de datos que incluyen metadatos de imagen, información de categorías y listas de anotaciones. Cada máscara binaria generada por *SAM-2* se transforma al formato *RLE* mediante las herramientas correspondientes, calculando automáticamente el área de cada instancia y asignando identificadores únicos secuenciales. El sistema exporta los resultados como archivos *JSON* organizados por clase experimental, manteniendo la trazabilidad completa desde las imágenes originales hasta las anotaciones finales.

## 4. Extracción de Subcubos Hiperespectrales

El último componente del flujo se encarga de la extracción de sub-cubos hiperespectrales radiométricamente corregidos a partir de las detecciones validadas en la fase anterior. Este módulo implementa un procesamiento sofisticado que combina corrección radiométrica, mapeo geométrico y extracción volumétrica para generar datos hiperespectrales de alta calidad correspondientes a cada higo individual detectado.

La corrección radiométrica constituye un paso crítico para garantizar la calidad de los datos hiperespectrales, eliminando efectos de iluminación y variaciones instrumentales que podrían comprometer el análisis posterior. El

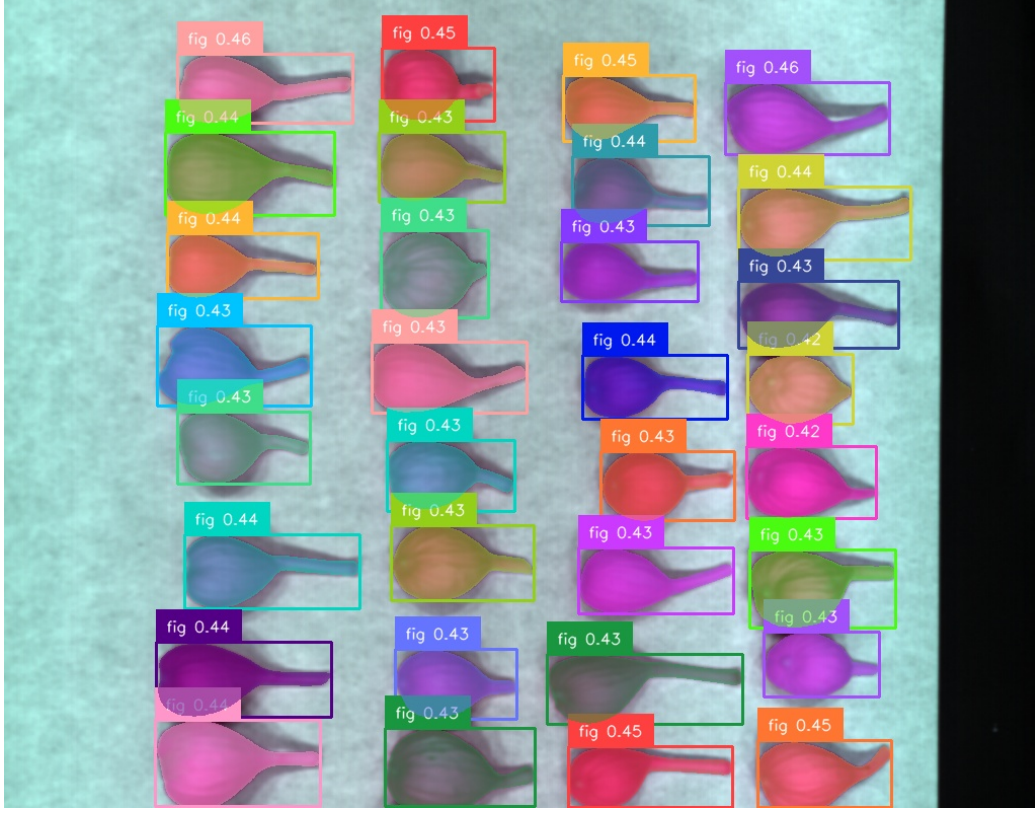


Figura 3.1: Ejemplo de detección y segmentación de higos utilizando *Grounding DINO* y *SAM-2*. La imagen muestra las cajas delimitadoras generadas por *Grounding DINO* y las máscaras de segmentación producidas por *SAM-2*.

proceso utiliza las referencias oscura y blanca capturadas simultáneamente con cada imagen hiperespectral, aplicando la ecuación estándar de corrección:

$$R = \frac{RAW - DARK}{WHITE - DARK} \quad (3.1)$$

donde  $R$  representa la reflectancia corregida,  $RAW$  los datos espectrales en bruto,  $DARK$  la referencia oscura y  $WHITE$  la referencia blanca. La implementación incluye el tratamiento robusto de casos especiales, como la prevención de divisiones por cero en regiones donde las referencias oscura y blanca presentan valores idénticos, situación que puede ocurrir en áreas de muy baja reflectancia.

La extracción geométrica de sub-cubos requiere una transformación precisa desde el espacio de coordenadas  $RGB$ , donde se realizaron las detecciones, al espacio hiperespectral correspondiente. Esta conversión considera las posibles diferencias en resolución espacial entre las imágenes  $RGB$  derivadas y los



Figura 3.2: Ejemplo de la representación en falso color *RGB* de un higo individual extraído del cubo hiperspectral. Las bandas espectrales utilizadas para la visualización son 220 (Rojo), 113 (Verde) y 78 (Azul).

cubos hiperspectrales originales, implementando técnicas de mapeo que preservan la correspondencia espacial exacta. El sistema valida geométricamente cada región de interés para asegurar que los subcubos extraídos no excedan los límites físicos del cubo hiperspectral, evitando errores de indexación y garantizando la integridad de los datos espectrales.

El proceso de extracción volumétrica utiliza técnicas de división tridimensional optimizadas para mantener la estructura espectral completa de cada región de interés. Los subcubos resultantes preservan las 448 bandas espectrales originales junto con la resolución espacial correspondiente a cada detección, manteniendo la información espectral íntegra necesaria para el análisis posterior. La organización del almacenamiento sigue una estructura jerárquica específica, con subdirectorios organizados por clase experimental (C0, C1, C2, C3). Cada subcubo se almacena en formato *.npy* de *NumPy* siguiendo una nomenclatura sistemática que incluye clase, timestamp y número de instancia, facilitando tanto el acceso eficiente como la trazabilidad completa mientras preserva la precisión numérica de punto flotante y optimiza los tiempos de carga durante el entrenamiento de modelos.

### 3.3.4. Resultados

La ejecución completa de la primera fase genera un conjunto estructurado de elementos que constituyen la base fundamental para las fases posteriores del proyecto. Las anotaciones *COCO* resultantes comprenden archivos *JSON* organizados por clase experimental, cada uno conteniendo metada-



tos completos de detección que incluyen coordenadas de cuadros delimitadores, máscaras de segmentación en formato *RLE* y metadatos temporales extraídos automáticamente del sistema de nomenclatura implementado. Esta organización sistemática permite mantener la trazabilidad completa desde las imágenes originales hasta las detecciones finales, facilitando tanto la validación manual como el procesamiento automatizado en etapas subsecuentes.

```

{
  "info": {
    "description": "Fig detection and segmentation dataset",
    "version": "1.0",
    "year": 2024,
    "contributor": "Hyperspectral Analysis Pipeline",
    "date_created": "2024-01-15"
  },
  "licenses": [],
  "images": [
    {
      "id": 1,
      "width": 800,
      "height": 1024,
      "file_name": "C0_2023-07-17_10-15-30_001.png",
      "date_captured": "2023-07-17T10:15:30"
    }
  ],
  "annotations": [
    {
      "id": 1,
      "image_id": 1,
      "category_id": 1,
      "segmentation": {
        "size": [1024, 800],
        "counts": "nXh04M3M2N2N10101N2N2N10101N2N..."
      },
      "area": 12485,
      "bbox": [245.3, 412.7, 128.4, 97.2],
      "iscrowd": 0
    }
  ],
  "categories": [
    {
      "id": 1,
      "name": "fig",
      "supercategory": "fruit"
    }
  ]
}

```

Figura 3.3: Ejemplo de anotación en formato COCO para la clase 0.

### 3.3.5. Desafíos y Observaciones Técnicas

Durante la implementación, se identificaron y resolvieron varios desafíos técnicos que proporcionaron valiosas lecciones para el desarrollo del proyecto. El primer y más significativo desafío encontrado fue determinar las versiones correctas de *PyTorch* y sus dependencias relacionadas (*torchvision* y *torchaudio*) que fueran compatibles tanto con *Grounding DINO* como con *SAM-2*. Ambos modelos requieren versiones específicas del framework que no siempre coinciden, especialmente considerando las actualizaciones frecuentes en el ecosistema de aprendizaje profundo. La solución final involucró el análisis detallado de los requisitos de compatibilidad de cada modelo y la identificación de una versión común de *PyTorch* 2.1.2 con soporte *CUDA* 11.8 que proporcionara estabilidad y rendimiento óptimo para ambos componentes.

La optimización de la memoria de la *GPU* constituyó otro reto importante, dado que el procesamiento conjunto de *Grounding DINO* y *SAM-2* requirió una implementación cuidadosa de contextos autocast para prevenir desbordamientos de memoria. La solución implementada aplica precisión mixta de forma selectiva [49]: utiliza `torch.autocast` con `dtype=torch.bfloat16` únicamente para *SAM-2*, mientras mantiene precisión completa para *Grounding DINO*, equilibrando eficiencia computacional con calidad de inferencia. La elección del formato `bfloat16` se fundamenta en su diseño específico para aplicaciones de aprendizaje profundo, proporcionando un rango dinámico superior a `float16` tradicional [50]. Adicionalmente, se configuró el uso de *TF32* cuando está disponible en hardware compatible para optimizar las operaciones de multiplicación de matrices.

## 3.4. Selección de Bandas

### 3.4.1. Objetivo de la Fase

La segunda fase del proyecto se centra en la optimización de la selección de bandas espectrales mediante la implementación de un algoritmo genético que identifica automáticamente las tres bandas más informativas del cubo hiperspectral. El objetivo principal es reducir la dimensionalidad de los datos hiperspectrales de manera inteligente, preservando la información espectral crítica para la clasificación de contaminación por aflatoxinas mientras se optimiza la eficiencia computacional del entrenamiento de redes neuronales.

La reducción de 448 bandas espectrales a una combinación RGB de tres bandas estratégicamente seleccionadas permite aprovechar arquitecturas de

CNN preentrenadas diseñadas para imágenes RGB convencionales, facilitando el uso de técnicas de aprendizaje por transferencia mediante ajuste fino selectivo [28] con modelos robustos como *ResNet-50* [?] sin sacrificar la capacidad discriminativa del sistema.

### 3.4.2. Herramientas y Tecnologías Empleadas

La implementación de esta fase integra algoritmos evolutivos de optimización con técnicas de aprendizaje profundo, creando un sistema híbrido que combina la exploración global de los algoritmos genéticos con la capacidad de generalización de las redes neuronales convolucionales.

#### DEAP (Distributed Evolutionary Algorithms in Python)

*DEAP* [40] es un framework innovador de computación evolutiva diseñado específicamente para el prototipado rápido y la evaluación eficiente de ideas en el ámbito de la optimización bio-inspirada. A diferencia de otros frameworks tradicionales que imponen limitaciones mediante tipos predefinidos, *DEAP* adopta una filosofía de diseño que prioriza la flexibilidad y la transparencia, permitiendo a los desarrolladores crear tipos de datos apropiados, personalizar inicializadores según sus necesidades específicas y seleccionar operadores de manera explícita y fundamentada.

En el contexto específico de este proyecto, *DEAP* facilita la implementación de un algoritmo genético especializado para la selección de bandas hiperespectrales mediante la creación de tipos personalizados que representan combinaciones de índices espectrales, operadores de cruce y mutación que respetan las restricciones del dominio (rango  $[0, 447]$ ), y estrategias de selección por torneo optimizadas para el problema de clasificación.

#### ResNet-50

ResNet-50 [51] constituye la arquitectura base empleada para evaluar la aptitud de cada individuo en el algoritmo genético. Esta red neuronal convolucional profunda introduce el concepto de *residual connections*, que permiten entrenar de forma estable modelos sustancialmente más profundos al mitigar problemas de desvanecimiento del gradiente. Con sus 50 capas, *ResNet-50* logra un equilibrio entre profundidad y eficiencia computacional, ofreciendo un rendimiento robusto en tareas de clasificación de imágenes. Al estar preentrenada en el extenso conjunto de datos ImageNet [52], proporciona un

punto de partida sólido para el aprendizaje por transferencia, lo que permite aprovechar representaciones visuales generales y adaptarlas a la tarea específica de clasificación de imágenes *RGB* derivadas de las combinaciones de bandas espectrales seleccionadas por el algoritmo evolutivo.

### 3.4.3. Flujo de Procesamiento

El algoritmo genético implementado utiliza una arquitectura evolutiva estándar, adaptada específicamente para abordar el problema de selección de bandas hiperespectrales. Cada iteración del proceso evolutivo integra una evaluación basada en aprendizaje profundo, optimizando así la selección de bandas de manera eficiente.

#### 1. Representación y Inicialización

Cada individuo en la población se representa mediante un vector de tres enteros en el rango  $[0, 447]$ , donde cada posición del vector corresponde a una banda espectral específica: el primer entero representa la banda asignada al canal *R* (rojo), el segundo entero al canal *G* (verde), y el tercer entero al canal *B* (azul). Estas bandas seleccionadas se combinan para formar la imagen *RGB*. La población inicial se genera aleatoriamente con un tamaño de 20 individuos, valor determinado mediante experimentación empírica para equilibrar diversidad poblacional con eficiencia computacional. La inicialización uniforme garantiza la exploración inicial del espacio completo de 448 bandas disponibles, evitando sesgos hacia regiones específicas del espectro electromagnético.

#### 2. Función de Evaluación

La evaluación de la aptitud constituye el componente más computacionalmente intensivo del algoritmo, requiriendo el entrenamiento mediante ajuste fino parcial de una red *ResNet-50* para cada individuo evaluado. El proceso comienza con la construcción de imágenes *RGB* utilizando las tres bandas especificadas por el individuo, seguido de la creación de cargadores de datos de *PyTorch* para los conjuntos de entrenamiento y prueba.

El modelo *ResNet-50* preentrenado en *ImageNet* se adapta específicamente para el problema de clasificación de cuatro clases mediante una estrategia de aprendizaje por transferencia basada en ajuste fino selectivo. La arquitectura implementa las siguientes modificaciones: (1) congelación de todas las capas desde la entrada hasta `layer3` (inclusive), manteniendo los pesos

preentrenados para la extracción de características de bajo y medio nivel; (2) liberación de los parámetros de **layer4**, la capa convolucional más profunda, permitiendo la adaptación de características de alto nivel específicas para la clasificación de higos con diferentes niveles de contaminación; y (3) reemplazo completo de la capa de clasificación final (**fc**) por una nueva capa lineal con 4 unidades de salida correspondientes a las clases *C0*, *C1*, *C2* y *C3*, inicializada aleatoriamente.

Esta configuración resulta en un modelo con aproximadamente 2.3 millones de parámetros entrenables de los 25.6 millones totales, concentrando el aprendizaje en las representaciones más específicas del dominio mientras preserva las características generales aprendidas en *ImageNet*. El entrenamiento se ejecuta durante 50 épocas utilizando el optimizador *Adam* con una tasa de aprendizaje de 0.001, aplicando técnicas de aumento de datos que incluyen volteos horizontales y verticales aleatorios, rotaciones de hasta 15 grados, y normalización estándar de *ImageNet*. La aptitud final del individuo se define como la precisión de prueba alcanzada en la última época, proporcionando una medida directa de la capacidad clasificatoria de la combinación de bandas espectrales seleccionada.

Para optimizar la eficiencia computacional, se implementó un sistema de caché que almacena los resultados de evaluaciones previas, evitando el reentrenamiento de combinaciones de bandas ya evaluadas en generaciones anteriores.

### 3. Operadores Genéticos

El algoritmo implementa operadores genéticos especializados que respetan las restricciones del dominio espectral y aprovechan las características intrínsecas de la información hiperespectral. La selección de estos operadores se fundamenta en el principio de que la información espectral relevante para la clasificación tiende a concentrarse en bandas espectralmente adyacentes debido a la correlación espacial natural entre longitudes de onda vecinas en el espectro electromagnético.

**Operador de Cruce:** El cruce utiliza una variante modificada del operador de mezcla (*blend crossover*) que combina linealmente los valores de los padres. La fórmula utilizada es:

$$hijo_i = (1 - \gamma) \cdot padre1_i + \gamma \cdot padre2_i \quad (3.2)$$

donde  $\gamma$  se genera aleatoriamente en cada posición. Este operador resulta

particularmente apropiado para la selección de bandas hiperespectrales porque produce descendientes cuyos índices de banda se mantienen en regiones espectrales intermedias entre los padres, preservando la localidad espectral y evitando saltos abruptos hacia bandas distantes que podrían no contener información correlacionada. Tras la aplicación del operador, se aplican restricciones de dominio para garantizar que los descendientes mantengan índices válidos en el rango  $[0, 447]$ .

**Operador de Mutación:** La mutación emplea un operador gaussiano con desviación estándar de 1.0 y probabilidad individual de 0.1, seguido de restricción al dominio válido. La elección de la mutación gaussiana con desviación estándar reducida ( $\sigma = 1.0$ ) está específicamente diseñada para introducir variaciones locales que exploren bandas espectralmente cercanas a las actuales, aprovechando el hecho de que bandas adyacentes en el espectro electromagnético típicamente contienen información complementaria y correlacionada. Esta estrategia de mutación conservativa evita perturbaciones drásticas que podrían llevar la búsqueda hacia regiones espectrales completamente diferentes y potencialmente menos informativas, manteniendo la continuidad espectral mientras permite la exploración gradual del espacio de soluciones.

**Selección y Elitismo:** La selección de padres utiliza torneos de tamaño 3, proporcionando una presión selectiva moderada que equilibra la explotación de buenas soluciones con la exploración de nuevas regiones del espacio de búsqueda. Se implementa elitismo con tamaño 1, garantizando la preservación de la mejor solución encontrada a través de las generaciones y asegurando que el algoritmo no pierda combinaciones de bandas de alta aptitud durante el proceso evolutivo.

#### 4. Configuración Experimental

El algoritmo se configura para ejecutar durante 50 generaciones con una población de 20 individuos, utilizando probabilidades de cruzamiento y mutación de 0.8 y 0.15 respectivamente. Estos parámetros se determinaron mediante experimentación previa para optimizar el balance entre exploración y explotación en el contexto específico del problema de selección de bandas.

Para garantizar la reproducibilidad y robustez estadística, se implementó un sistema de experimentos múltiples que permite la ejecución de varias corridas independientes del algoritmo, cada una con semillas aleatorias diferentes. Los resultados de cada experimento se almacenan automáticamente en ar-

chivos CSV que incluyen estadísticas generacionales completas y métricas detalladas del mejor individuo.

### 3.4.4. Resultados

Se ejecutaron diez experimentos independientes. Los resultados demuestran variaciones significativas en el rendimiento clasificatorio según las bandas seleccionadas.

La Tabla 3.1 presenta el rendimiento de cada experimento. Las precisiones de prueba oscilaron entre 0.78 y 0.84, con una variabilidad de 6 puntos porcentuales que subraya la importancia de la selección inteligente de bandas.

Tabla 3.1: Resumen de métricas de los experimentos de selección de bandas.

exp	test-acc	test-loss	class-accuracy	macro-f1	weighted-f1	bands
1	0.81	0.72	0.78	0.78	0.78	161-174-112
2	0.82	0.61	0.80	0.80	0.80	110-97-172
3	0.82	0.77	0.80	0.80	0.80	231-150-277
4	0.83	0.64	0.81	0.81	0.81	125-156-84
5	0.80	0.74	0.78	0.78	0.78	215-147-173
6	0.82	0.72	0.80	0.80	0.80	94-156-188
7	0.82	0.69	0.79	0.79	0.79	274-447-109
8	0.78	0.80	0.75	0.75	0.75	324-226-339
9	0.79	0.70	0.76	0.76	0.76	231-104-405
10	0.84	0.59	0.82	0.82	0.82	220-113-78

El Experimento 10 alcanzó el rendimiento superior con precisión de 0.84 utilizando las bandas [220, 113, 78], seguido por los Experimentos 2, 3, 4, 6 y 7 con precisión de 0.82. Los Experimentos 8 y 9 mostraron el rendimiento más bajo (0.78 y 0.79), indicando que sus combinaciones espectrales fueron menos efectivas para la discriminación de contaminación. La Figura 3.4 muestra las curvas de entrenamiento del experimento de mejor rendimiento, mientras que la Tabla 3.2 presenta sus métricas detalladas.

Tabla 3.2: Resultados del experimento 10 con las bandas seleccionadas 220, 113 y 78.

train-loss	train-acc	test-loss	test-acc
0.18	0.95	0.59	0.84

Las métricas  $F1$  macro mantuvieron consistencia con las tendencias de precisión (0.75-0.82), confirmando capacidades de discriminación equilibra-



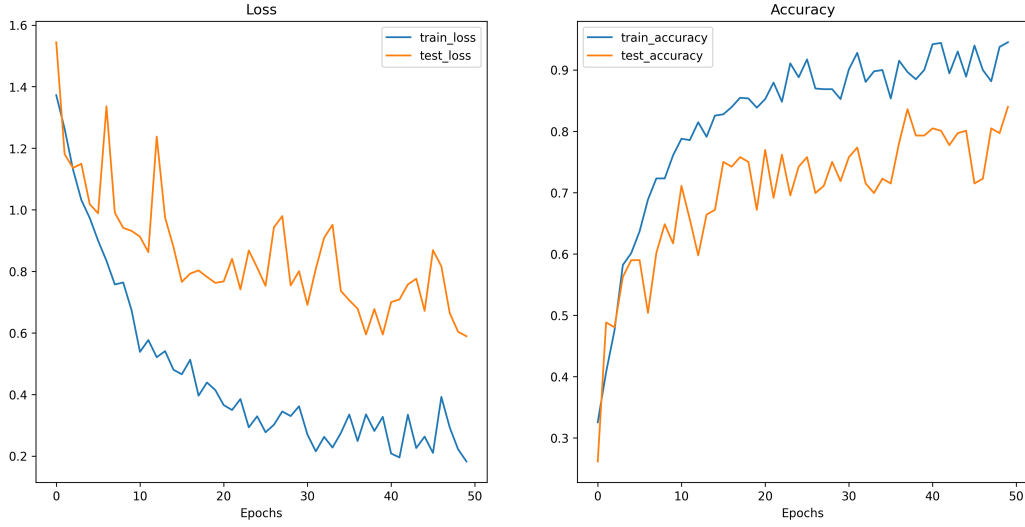


Figura 3.4: Curvas de precisión y pérdida del experimento 10 con las bandas seleccionadas 220, 113 y 78.

das entre las cuatro categorías de contaminación sin sesgos hacia clases específicas, como se detalla en la Tabla 3.3.

Tabla 3.3: Reporte de clasificación del experimento 10 con las bandas seleccionadas 220, 113 y 78.

	precision	recall	f1-score	support
0	0.80	0.84	0.82	56.00
1	0.86	0.79	0.82	56.00
2	0.82	0.82	0.82	57.00
3	0.80	0.82	0.81	57.00
accuracy	0.82	0.82	0.82	0.82
macro avg	0.82	0.82	0.82	226.00
weighted avg	0.82	0.82	0.82	226.00

El análisis de la evolución del fitness revela un patrón de convergencia consistente en todos los experimentos, con tres fases características. La **fase inicial** (generaciones 0-10) muestra mejora rápida del fitness promedio desde 0.69 hasta 0.79 con alta variabilidad poblacional. La **fase de convergencia** (generaciones 10-25) estabiliza el fitness promedio en 0.82-0.84 con reducción gradual de variabilidad. La **fase final** (generaciones 25-50) exhibe convergencia completa hacia el valor óptimo de 0.84, confirmando la estabilidad de las soluciones identificadas y la robustez del algoritmo evolutivo, como se

observa en la Figura 3.5.

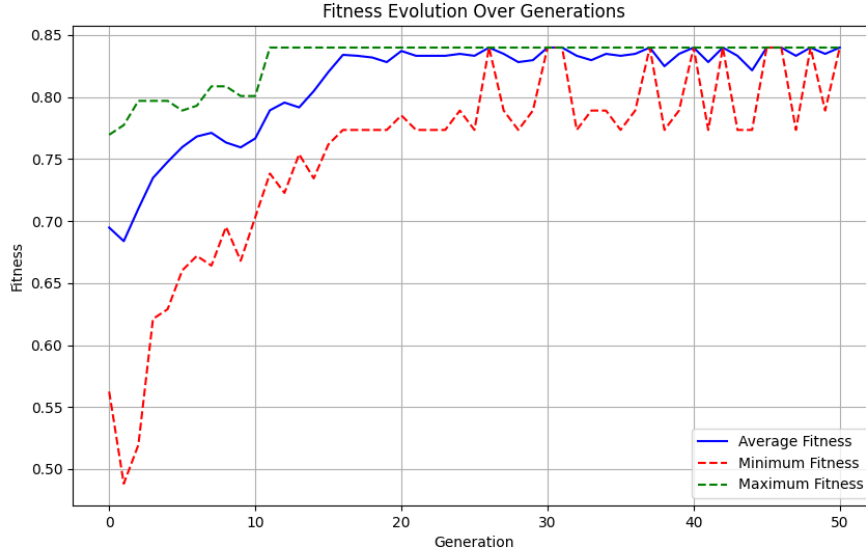


Figura 3.5: Curvas de evolución de la aptitud del experimento 10 con las bandas seleccionadas 220, 113 y 78.

El Experimento 10 representa la configuración de máximo rendimiento, utilizando las bandas 220, 113 y 78. El reporte de clasificación detallado (Tabla 3.3) muestra:

- **C0 (Saludables):** Precisión = 0.797, Recall = 0.839, F1 = 0.817
- **C1 (Baja contaminación):** Precisión = 0.863, Recall = 0.786, F1 = 0.822
- **C2 (Media contaminación):** Precisión = 0.825, Recall = 0.825, F1 = 0.825
- **C3 (Alta contaminación):** Precisión = 0.797, Recall = 0.825, F1 = 0.810

La precisión global alcanzó 81.9% con promedios macro de 0.820, 0.819 y 0.819 para precisión, recall y F1-score respectivamente. La matriz de confusión (Tabla 3.4) reveló tasas de detección verdadero positivo robustas: C0 (83.9%), C1 (78.6%), C2 (82.5%) y C3 (82.5%), con errores de clasificación distribuidos uniformemente sin sesgos sistemáticos.

Tabla 3.4: Matriz de confusión del experimento 10 con las bandas seleccionadas 220, 113 y 78.

	C0	C1	C2	C3
C0	47	2	1	6
C1	3	44	5	4
C2	6	2	47	2
C3	3	3	4	47

Los resultados demuestran la efectividad de la metodología híbrida que combina algoritmos genéticos con redes neuronales convolucionales para la selección automática de bandas espectrales. La convergencia consistente observada entre las generaciones 10-25 confirma la eficiencia del algoritmo evolutivo para identificar soluciones óptimas evitando la exploración exhaustiva de  $\binom{448}{3} \approx 1,5 \times 10^7$  combinaciones posibles.

### 3.4.5. Desafíos y Observaciones Técnicas

Durante la implementación de esta fase del proyecto, se identificaron varios desafíos técnicos significativos que impactaron tanto en los tiempos de desarrollo como en la metodología experimental. Estos desafíos proporcionan conocimientos valiosos para futuras implementaciones y optimizaciones del sistema.

Uno de los principales obstáculos técnicos encontrados se relacionó con la configuración de los cargadores de datos de PyTorch (`DataLoader`). Durante las pruebas iniciales, se observó que configurar el parámetro `num_workers` con valores superiores a 0 resultaba en tiempos de ejecución significativamente prolongados, contrario a las expectativas teóricas de paralelización. Este comportamiento aparentemente contradictorio se debe a varias consideraciones específicas del entorno de ejecución. En sistemas con limitaciones de memoria RAM o configuraciones específicas de hardware, la creación de múltiples procesos trabajadores puede introducir una sobrecarga significati-

va debido al costo de inicialización de procesos, transferencia de datos entre procesos padre e hijos, y sincronización. Esta sobrecarga puede superar los beneficios de la paralelización, especialmente cuando el conjunto de datos no es lo suficientemente grande o complejo como para justificar la distribución del trabajo.

Además, el uso de múltiples procesos trabajadores puede generar contención por recursos del sistema como acceso a disco, memoria compartida, y ancho de banda de memoria, lo que paradójicamente puede ralentizar el proceso de carga de datos. En el contexto de este proyecto, donde cada muestra requiere la carga y procesamiento de datos hiperespectrales de alta dimensionalidad (448 bandas espectrales), la contención de recursos se vuelve particularmente crítica. Este problema se ve agravado en entornos de servidores compartidos como este, donde múltiples usuarios ejecutan trabajos simultáneamente, ya que aunque la GPU pueda estar dedicada exclusivamente al proceso de entrenamiento, otros recursos críticos como CPU, memoria RAM, ancho de banda de memoria y sistema de I/O de disco pueden estar bajo demanda concurrente, incrementando significativamente la latencia de operaciones de carga de datos paralelas. La solución adoptada fue mantener `num_workers=0`, configuración que fuerza la carga secuencial de datos en el proceso principal. Aunque esta configuración elimina el paralelismo en la carga de datos, resultó en tiempos de entrenamiento significativamente más eficientes para las características específicas del conjunto de datos y la infraestructura utilizada.

La definición de la arquitectura óptima del modelo constituyó un proceso iterativo y computacionalmente intensivo que requirió varias semanas de experimentación sistemática. Este proceso involucró la exploración de múltiples configuraciones arquitectónicas, cada una requiriendo ciclos completos de entrenamiento para su evaluación. Se evaluaron diferentes estrategias de congelación de capas en la red ResNet-50 preentrenada, incluyendo: (1) congelación completa de la estructura base con entrenamiento únicamente de la capa clasificadora final; (2) liberación progresiva de capas desde la capa de clasificación hacia capas convolucionales más profundas; y (3) configuraciones híbridas que combinan congelación selectiva con tasas de aprendizaje diferenciadas. La configuración final, que congela capas hasta `layer3` inclusive y libera `layer4` y la capa `fc`, emergió como el equilibrio óptimo entre capacidad de adaptación al dominio específico y preservación de características preentrenadas útiles.

Se exploraron arquitecturas alternativas incluyendo variantes de Effi-

cientNet, DenseNet, y configuraciones personalizadas de ResNet con diferentes profundidades. Cada arquitectura requirió ajustes específicos de hiperparámetros, estrategias de regularización, y técnicas de aumento de datos, resultando en un proceso experimental extensivo que, aunque costoso temporalmente, fue fundamental para identificar la configuración óptima.

## 3.5. Análisis Hiperespectral mediante Transformadas Wavelet

### 3.5.1. Objetivo de la Fase

La tercera fase del proyecto implementa una metodología alternativa para el análisis de imágenes hiperespectrales mediante la aplicación de transformadas *wavelet* en regiones espacialmente localizadas del higo. Esta aproximación sigue la línea de investigación establecida por Cruz-Carrasco et al. [10], quienes demostraron la efectividad del análisis hiperespectral combinado con transformadas *wavelet* para la detección de *Aspergillus flavus* en higos. Sin embargo, mientras que el estudio de referencia realizó el análisis a nivel de píxel individual, esta fase extiende la metodología considerando regiones espaciales más amplias que capturan información contextual y variabilidad espacial local.

El objetivo principal es explorar la capacidad de las transformadas *wavelet* continuas (CWT) y discretas (DWT) para extraer características espectrales discriminativas a partir de regiones de  $32 \times 32$  píxeles subdivididas en parches de  $4 \times 4$  píxeles, permitiendo la clasificación efectiva de diferentes niveles de contaminación por aflatoxinas. Esta aproximación regional, en contraste con el análisis puntual por píxeles, permite capturar patrones espaciales de heterogeneidad espectral que pueden indicar diferentes etapas del desarrollo fúngico y la distribución espacial de la contaminación dentro del tejido del higo.

Esta aproximación se fundamenta en la hipótesis de que las transformadas *wavelet* pueden capturar información espectral-temporal crítica que métodos tradicionales de procesamiento espectral podrían no detectar. A diferencia de las fases anteriores que operan sobre imágenes *RGB* derivadas o selecciones optimizadas de bandas espectrales, esta metodología procesa directamente la información espectral completa mediante análisis tiempo-frecuencia, preservando tanto las características espectrales como sus variaciones temporales a lo largo del espectro electromagnético.

### 3.5.2. Herramientas y Tecnologías Empleadas

La implementación de esta fase integra técnicas avanzadas de procesamiento de señales con arquitecturas de aprendizaje profundo especializadas, combinando análisis *wavelet* con redes neuronales convolucionales para la clasificación automatizada de muestras.

#### PyWavelets

*PyWavelets* [53] es una biblioteca de Python de código abierto que implementa de manera eficiente y numéricamente estable una amplia gama de transformadas *wavelet*, tanto en su versión discreta como continua. Ofrece soporte para transformadas multiescala, transformadas inversas, y herramientas de análisis como filtrado, compresión y descomposición de señales. La biblioteca incluye múltiples familias de *wavelets* (*Daubechies*, *Morlet*, *Haar*, *Biorthogonal*, entre otras), lo que permite seleccionar la función base más adecuada según la aplicación. En el marco de este proyecto, *PyWavelets* se emplea para transformar firmas espectrales unidimensionales en representaciones tiempo-frecuencia bidimensionales (escalogramas), que capturan de forma localizada tanto las características frecuenciales como su posición en el dominio espectral.

#### DenseNet-121

La arquitectura *DenseNet-121* [54] se caracteriza por el uso de conexiones densas entre capas, en las cuales cada capa recibe como entrada no solo la salida de la capa inmediatamente anterior, sino también los mapas de características de todas las capas previas, lo que fomenta una reutilización eficiente de la información aprendida. Esta estrategia de conectividad permite mitigar el problema del desvanecimiento del gradiente, favoreciendo un entrenamiento más estable y profundo, al tiempo que reduce la redundancia de parámetros en comparación con otras arquitecturas tradicionales como *ResNet*, gracias a la mayor eficiencia en el flujo de información. *DenseNet-121*, con sus 121 capas organizadas en bloques densos e intercaladas con capas de transición que controlan la dimensionalidad y la complejidad computacional, logra un equilibrio entre profundidad, capacidad de representación y eficiencia, lo que la convierte en una de las arquitecturas más utilizadas en tareas de clasificación y extracción de características en el dominio del aprendizaje profundo.

### 3.5.3. Flujo de Procesamiento

El flujo de procesamiento implementa una metodología sistemática que combina extracción espacial de regiones de interés, análisis espectral mediante transformadas *wavelet*, y clasificación mediante aprendizaje profundo.

#### 1. Extracción de Parches Espectrales

El proceso comienza con la identificación automática de regiones óptimas de  $32 \times 32$  píxeles en cada imagen hiperespectral. El algoritmo implementa una estrategia de búsqueda espacial que localiza la región centrada verticalmente y posicionada horizontalmente para minimizar la presencia de píxeles de fondo. Esta selección automatizada garantiza que los parches extraídos contengan exclusivamente información espectral del tejido del higo, eliminando interferencias del fondo de la imagen que podrían introducir ruido en el análisis posterior.

Una vez identificada la región óptima de  $32 \times 32$  píxeles, se procede a la subdivisión sistemática en una grilla regular de 64 sub-parches de  $4 \times 4$  píxeles cada uno. Para cada sub-parche de  $4 \times 4$  píxeles, se extrae la firma espectral promedio calculando la media aritmética de los valores espectrales a través de las dimensiones espaciales. Este procedimiento de promediado espacial genera 64 firmas espectrales representativas, cada una de 448 bandas espectrales, que capturan la información espectral característica de regiones espacialmente localizadas dentro del higo individual.

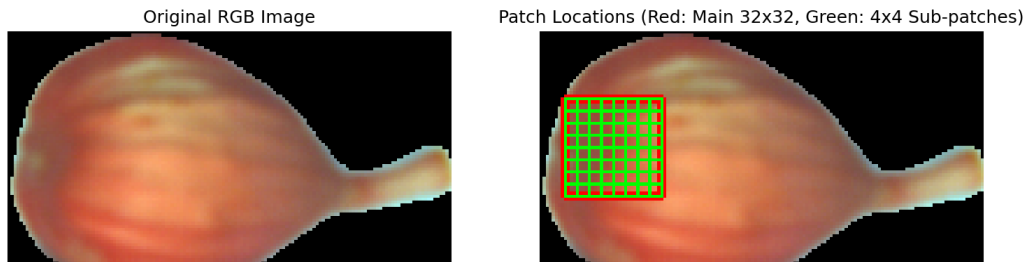


Figura 3.6: Visualización de la región de  $32 \times 32$  píxeles extraída automáticamente del higo, subdividida en 64 parches de  $4 \times 4$  píxeles.

Esta metodología de promediado espacial permite reducir el ruido espectral inherente a nivel de píxel individual mientras preserva las características espectrales distintivas de cada región. Al calcular la media espectral de los 16 píxeles ( $4 \times 4$ ) que componen cada sub-parche, se obtiene una representación espectral más robusta y estable que facilita la identificación de patrones

espectrales asociados con diferentes grados de contaminación fúngica, como se ilustra en la Figura 3.7.

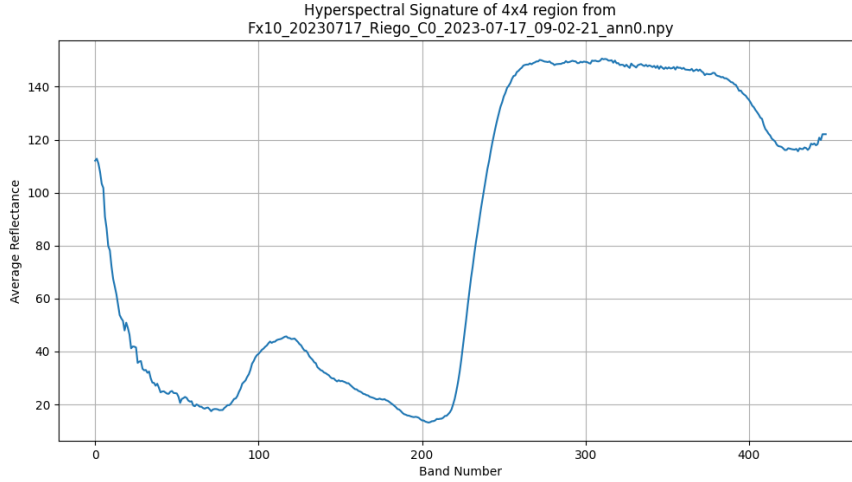


Figura 3.7: Ejemplo de firma espectral promedio extraída de un parche de  $4 \times 4$  píxeles. La gráfica muestra la reflectancia en función de las 448 bandas espectrales.

## 2. Transformada Wavelet Continua

La transformación espectral utiliza la Transformada Wavelet Continua (*CWT*) con la *wavelet* de *Morlet* como función base. Esta elección se fundamenta en las propiedades óptimas de la *wavelet* de *Morlet* para análisis espectral: proporciona una resolución tiempo-frecuencia balanceada, mantiene fase constante, y exhibe características de localización espectral apropiadas para señales hiperespectrales. La *CWT* se aplica a cada firma espectral de 448 bandas utilizando 64 escalas logarítmicamente distribuidas, generando escalogramas de  $64 \times 448$  píxeles que representan la distribución tiempo-frecuencia de la información espectral.

Los escalogramas resultantes capturan tanto características espectrales globales como variaciones localizadas en frecuencia, proporcionando una representación rica en información que preserva patrones espectrales discriminativos para la detección de contaminación por aflatoxinas. La selección de 64 escalas permite una cobertura completa del rango espectral mientras mantiene una resolución suficiente para detectar características espectrales finas asociadas con cambios bioquímicos inducidos por el desarrollo fúngico, como se muestra en la Figura 3.8.



### 3. Preparación de Datos para CNN

Los 64 escalogramas generados por cada parche de  $32 \times 32$  píxeles se organizan como imágenes individuales para el entrenamiento de la red neuronal convolucional. Esta estrategia de aumentado de datos resulta en una expansión significativa del conjunto de datos: cada imagen hiperspectral original genera 64 instancias de entrenamiento, multiplicando efectivamente el tamaño del conjunto de datos disponible para el aprendizaje supervisado.

Los escalogramas se normalizan utilizando estadísticas estándar de *ImageNet* para aprovechar las ventajas del aprendizaje por transferencia. La organización del conjunto de datos sigue una estructura jerárquica estándar con directorios separados para entrenamiento y prueba, manteniendo la distribución de clases balanceada (*C0*, *C1*, *C2*, *C3*) y preservando la trazabilidad desde las imágenes hiperspectrales originales hasta los escalogramas individuales.

### 4. Arquitectura y Entrenamiento de CNN

La red *DenseNet-121* se adapta específicamente para la clasificación de cuatro clases mediante la modificación de la capa clasificadora final. La arquitectura implementa una cabeza clasificadora personalizada que incluye capas lineales con normalización por lotes, funciones de activación *ReLU*, y regularización mediante *dropout* para prevenir sobreajuste. La configuración final comprende: una capa lineal de 1024 a 128 unidades con *ReLU* y normalización por lotes, seguida de *dropout* (0.4), una segunda capa lineal de 128 a 64 unidades con *ReLU* y *dropout* (0.3), y finalmente una capa de salida de 64 a 4 unidades correspondientes a las clases experimentales, como se detalla en la Tabla 3.5.

El entrenamiento se ejecuta durante 50 épocas utilizando el optimizador *Adam* con tasa de aprendizaje inicial de 0.001 y un planificador *ReduceLROnPlateau* que reduce la tasa de aprendizaje cuando la pérdida de validación se estanca. El modelo utiliza función de pérdida de entropía cruzada y técnicas de aumento de datos que incluyen transformaciones estocásticas apropiadas para imágenes de escalogramas, preservando las características tiempo-frecuencia mientras introducen variabilidad para mejorar la generalización.

Tabla 3.5: Configuración de la cabeza clasificadora personalizada para *DenseNet-121*.

Layer	Output Format	Training Settings
ReLU (1)	128	-
BatchNorm1d (2)	128	256
Dropout (3)	128	-
ReLU (5)	64	-
Dropout (6)	64	-
Linear (7)	4	260
Total	-	139972

### 3.5.4. Resultados

La metodología basada en transformadas wavelet demostró una efectividad notable para la clasificación de contaminación por aflatoxinas en muestras de higo. El modelo *DenseNet-121* entrenado alcanzó una precisión de clasificación de **87.43 %** en el conjunto de prueba, con métricas consistentes a través de las cuatro clases experimentales. La Figura 3.9 muestra la evolución del entrenamiento durante 50 épocas.

El análisis detallado por clases revela un rendimiento balanceado: *C0* (control sano) alcanzó precisión de 88.92 % y recall de 88.59 %, *C1* (baja contaminación) obtuvo precisión y recall de 88.26 %, *C2* (contaminación media) logró precisión de 85.24 % y recall de 84.49 %, mientras que *C3* (alta contaminación) registró precisión de 87.35 % y recall de 88.43 %. Los puntajes F1 correspondientes fueron 88.76 %, 88.26 %, 84.86 % y 87.89 % respectivamente, indicando un equilibrio efectivo entre precisión y recall en todas las categorías, como se presenta en la Tabla 3.6.

La matriz de confusión revela patrones de clasificación interpretables: las confusiones más frecuentes ocurren entre clases adyacentes (*C1-C2* y *C2-C3*), reflejando la progresión gradual de la contaminación por aflatoxinas. Específicamente, se observan 190 confusiones entre *C1* y *C2*, y 174 confusiones entre *C2* y *C3*. La clase control (*C0*) muestra la menor tasa de confusión con clases contaminadas, con solo 403 clasificaciones erróneas de un total de 3533 muestras, indicando que la metodología wavelet es efectiva para distinguir muestras sanas de contaminadas, como se detalla en la Tabla 3.7.

El entrenamiento del modelo se completó en 50 épocas, con un tiempo

Tabla 3.6: Reporte de clasificación del modelo basado en transformadas wavelet.

	precision	recall	f1-score	support
C0	0.89	0.89	0.89	3533.00
C1	0.88	0.88	0.88	3559.00
C2	0.85	0.84	0.85	3636.00
C3	0.87	0.88	0.88	3623.00
accuracy	0.87	0.87	0.87	0.87
macro avg	0.87	0.87	0.87	14351.00
weighted avg	0.87	0.87	0.87	14351.00

Tabla 3.7: Matriz de confusión del modelo basado en transformadas wavelet.

	C0	C1	C2	C3
C0	3130	100	166	137
C1	102	3141	192	124
C2	171	190	3072	203
C3	117	128	174	3204

total de entrenamiento de aproximadamente 3.1 horas en una GPU NVIDIA A100. La convergencia del modelo se observa claramente en las curvas de entrenamiento mostradas en la Figura 3.9, donde tanto la pérdida como la precisión se estabilizan después de las primeras 20 épocas, indicando un aprendizaje efectivo sin evidencia significativa de sobreajuste.

### 3.5.5. Desafíos y Observaciones Técnicas

Durante la implementación, se identificó un desafío técnico significativo relacionado con la estabilidad del entrenamiento de la red neuronal convolucional. El principal obstáculo encontrado fue la presencia de fluctuaciones extremas e inconsistentes en las métricas de validación (pérdida de prueba y precisión de prueba) durante el proceso de entrenamiento, mientras que las métricas de entrenamiento mostraban una progresión estable y monotónica-

mente decreciente.

Este comportamiento anómalo se caracterizó por oscilaciones abruptas y de gran amplitud en la pérdida de validación, con valores que variaban drásticamente entre épocas consecutivas. Por ejemplo, se observaron saltos desde una pérdida de prueba de 0.5782 en la época 26 hasta 2.0304 en la época 27, o fluctuaciones desde 0.4761 en la época 44 hasta 3.3955 en la época 46. De manera similar, la precisión de validación experimentó variaciones extremas, oscilando entre valores tan altos como 0.8413 y tan bajos como 0.3364 en épocas próximas. La Figura 3.10 ilustra claramente este comportamiento errático en las curvas de entrenamiento iniciales.

Este fenómeno es indicativo de inestabilidad en el proceso de optimización, comúnmente asociado con tasas de aprendizaje inadecuadas que provocan que el optimizador *Adam* realice actualizaciones de parámetros demasiado agresivas. Las fluctuaciones sugieren que el modelo estaba "saltando" entre diferentes regiones del espacio de pérdida sin converger hacia un mínimo estable, particularmente problemático en arquitecturas profundas como *DenseNet-121* cuando se aplican a dominios especializados como el análisis de escalogramas wavelet.

La solución implementada consistió en la incorporación de un planificador de tasa de aprendizaje adaptativo (*ReduceLROnPlateau*) que monitorea la pérdida de validación y reduce automáticamente la tasa de aprendizaje cuando se detecta estancamiento en la mejora del modelo. Esta estrategia permite que el entrenamiento comience con una tasa de aprendizaje más alta para exploración inicial, seguida de reducciones progresivas que facilitan la convergencia fina hacia mínimos locales estables. El planificador se configuró con un factor de reducción de 0.5, paciencia de 5 épocas, y un umbral mínimo de mejora de 0.01, parámetros que demostraron ser efectivos para estabilizar el entrenamiento y lograr la convergencia consistente observada en los resultados finales.

La implementación del planificador no solo eliminó las fluctuaciones erráticas sino que también mejoró significativamente la capacidad de generalización del modelo, como se evidencia en las curvas de entrenamiento estables mostradas en la Figura 3.9 y la precisión final de 87.43 % alcanzada en el conjunto de prueba.

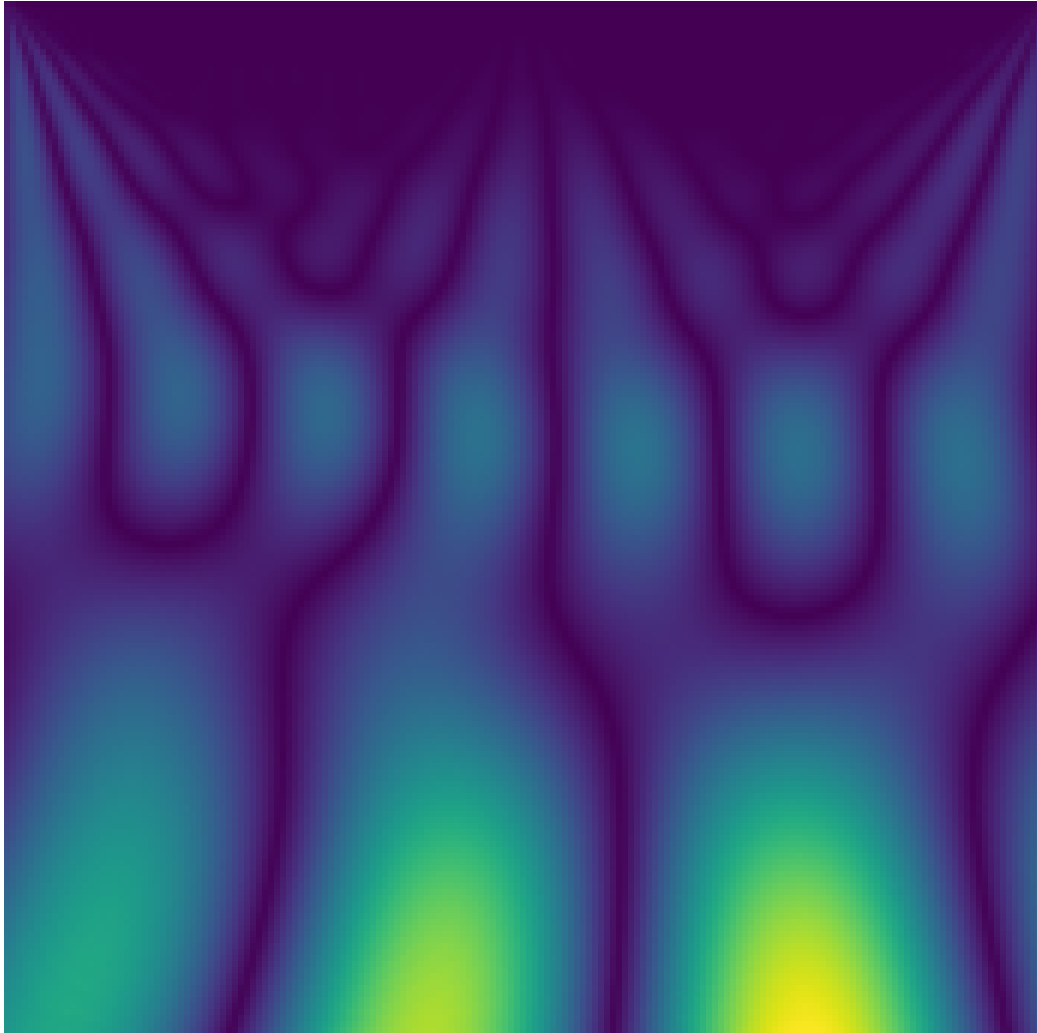


Figura 3.8: Ejemplo de escalograma generado por la Transformada Wavelet Continua ( $CWT$ ) utilizando la *wavelet* de *Morlet*.

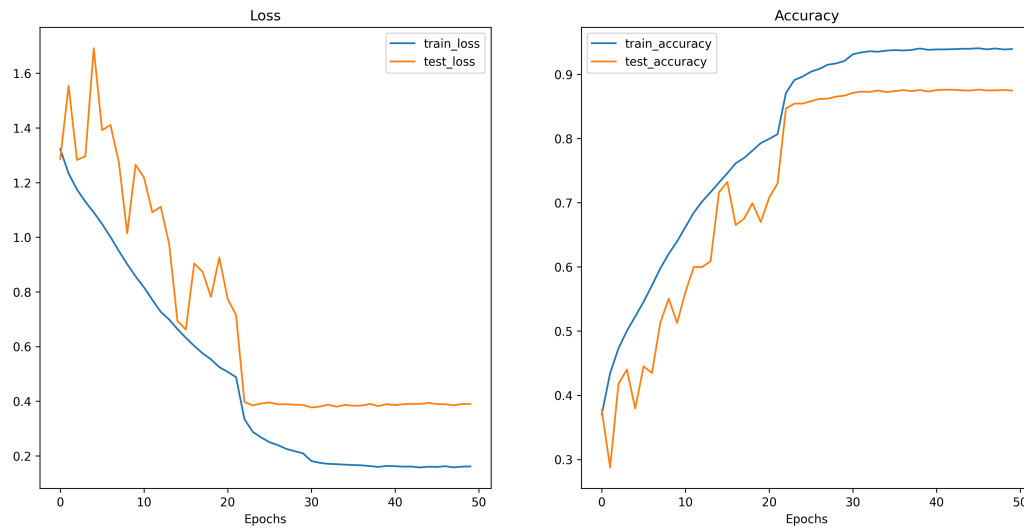


Figura 3.9: Curvas de entrenamiento del modelo *DenseNet-121* para clasificación de escalogramas wavelet. Se muestran la evolución de la pérdida y precisión durante 50 épocas de entrenamiento.

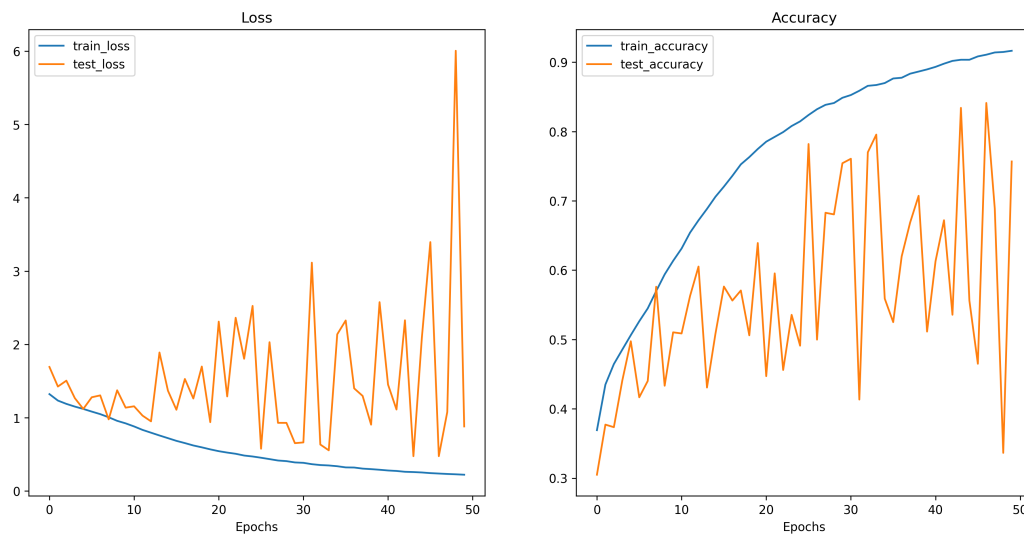


Figura 3.10: Comportamiento errático inicial de las métricas de validación, mostrando fluctuaciones extremas en la pérdida de prueba y precisión de prueba durante el entrenamiento sin planificador.

## Capítulo 4

# Resultados

La selección evolutiva de bandas con ResNet-50 consolidó un *pipeline* RGB capaz de alcanzar una precisión global del 81.9 %, con métricas *macro* equilibradas y detecciones verdaderas positivas cercanas al 80 % en las cuatro clases, lo que confirma la viabilidad de reducir el cubo hiperespectral a combinaciones tricanal para clasificación automática.

Por su parte, la caracterización tiempo-frecuencia mediante transformadas *wavelet* y DenseNet-121 elevó la precisión al 87.43 %, manteniendo la coherencia entre precisión y *recall* por clase y concentrando los errores en transiciones progresivas de contaminación, lo que evidencia mayor sensibilidad a las variaciones bioquímicas del tejido.

El método *wavelet* aventaja al esquema RGB-genético en 5.5 puntos porcentuales de precisión absoluta (87.43 % frente a 81.9 %), además de reducir las confusiones con la clase sana y distribuir los errores según la progresión natural de la infección. Esta mejora se atribuye a la riqueza de los escalogramas, que encapsulan patrones multiescala que permanecen latentes cuando se colapsa el cubo a tres bandas.

No obstante, el enfoque RGB sigue ofreciendo una solución más liviana, con menor demanda computacional y una interpretación directa en formato de imagen convencional.

## Capítulo 5

### Conclusiones y Trabajo Futuro



# Bibliografía

- [1] R. Khan, F. Anwar, and F. M. Ghazali, “A comprehensive review of mycotoxins: Toxicology, detection, and effective mitigation approaches,” *Heliyon*, vol. 10, no. 8, p. e28361, 2024.
- [2] IARC Working Group on the Evaluation of Carcinogenic Risks to Humans, *Chemical Agents and Related Occupations. AFLATOXINS*. IARC Monographs on the Evaluation of Carcinogenic Risks to Humans, No. 100F, Lyon, France: International Agency for Research on Cancer, 2012.
- [3] B. Tai, J. Chang, Y. Liu, and F. Xing, “Recent progress of the effect of environmental factors on aspergillus flavus growth and aflatoxins production on foods,” *Food Quality and Safety*, vol. 4, pp. 21–28, 02 2020.
- [4] T. Ahmad, S. Wang, and Y. Liu, “Aspergillus flavus and aflatoxins (3rd edition),” *Toxins (Basel)*, vol. 17, p. 326, June 2025.
- [5] P. Bakewell-Stone, “Ficus carica (common fig),” *CABI Compendium*, 2022.
- [6] Ministerio de Agricultura, Pesca y Alimentación, “Encuesta sobre superficies y rendimientos de cultivos (esyrce) 2023,” tech. rep., Ministerio de Agricultura, Pesca y Alimentación, España, 2023.
- [7] A. Khan, A. D. Vibhute, S. Mali, and C. Patil, “A systematic review on hyperspectral imaging technology with a machine and deep learning methodology for agricultural applications,” *Ecological Informatics*, vol. 69, p. 101678, 2022.
- [8] I. Cisternas, I. Velásquez, A. Caro, and A. Rodríguez, “Systematic literature review of implementations of precision agriculture,” *Computers and Electronics in Agriculture*, vol. 176, p. 105626, 2020.
- [9] A. Signoroni, M. Savardi, A. Baronio, and S. Benini, “Deep learning meets hyperspectral image analysis: A multidisciplinary review,” *Journal of Imaging*, vol. 5, no. 5, 2019.

- [10] C. Cruz-Carrasco, J. Díaz-Álvarez, F. Chávez de la O, A. Sánchez-Venegas, and J. Villegas Cortez, “Detection of aspergillus flavus in figs by means of hyperspectral images and deep learning algorithms,” *AgriEngineering*, vol. 6, no. 4, pp. 3969–3988, 2024.
- [11] D. Hong, N. Yokoya, J. Chanussot, J. Xu, and X. X. Zhu, “Learning to propagate labels on graphs: An iterative multitask regression framework for semi-supervised hyperspectral dimensionality reduction,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 158, pp. 35–49, 2019.
- [12] R. Sahoo, S. Ray, and M. R., “Hyperspectral remote sensing of agriculture,” *Current science*, vol. 108, pp. 848–859, 03 2015.
- [13] J. Wieme, K. Mollazade, I. Malounas, M. Zude-Sasse, M. Zhao, A. Gowen, D. Argyropoulos, S. Fountas, and J. Van Beek, “Application of hyperspectral imaging systems and artificial intelligence for quality assessment of fruit, vegetables and mushrooms: A review,” *Biosystems Engineering*, vol. 222, pp. 156–176, 2022.
- [14] Z. Zhong, J. Li, Z. Luo, and M. Chapman, “Spectral-spatial residual network for hyperspectral image classification: A 3-d deep learning framework,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, pp. 847–858, 10 2017.
- [15] K. P. Ferentinos, “Deep learning models for plant disease detection and diagnosis,” *Computers and Electronics in Agriculture*, vol. 145, pp. 311–318, 2018.
- [16] M. Paoletti, J. Haut, J. Plaza, and A. Plaza, “Deep learning classifiers for hyperspectral imaging: A review,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 158, pp. 279–317, 2019.
- [17] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, “Deep learning-based classification of hyperspectral data,” *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, vol. 7, pp. 2094–2107, 06 2014.
- [18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” 2016.
- [19] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” 2014.

- [20] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” 2015.
- [21] V. Badrinarayanan, A. Kendall, and R. Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for image segmentation,” 2016.
- [22] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” 2017.
- [23] G. Van Rossum and F. L. Drake Jr, *Python tutorial*. Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995.
- [24] S. Behnel, R. Bradshaw, C. Citro, L. Dalcin, D. S. Seljebotn, and K. Smith, “Cython: The best of both worlds,” *Computing in Science & Engineering*, vol. 13, no. 2, pp. 31–39, 2011.
- [25] “Anaconda software distribution,” 2020.
- [26] Astral, “Uv: An extremely fast python package installer and resolver,” 2024. GitHub Repository. Disponible en: <https://github.com/astral-sh/uv>.
- [27] N. D. Matsakis and F. S. Klock II, “The rust language,” in *ACM SIGAda Ada Letters*, vol. 34, pp. 103–104, ACM, 2014.
- [28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, Curran Associates, Inc., 2019.
- [29] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. Fernández del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, “Array programming with NumPy,” *Nature*, vol. 585, p. 357–362, 2020.
- [30] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J.

- Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, Í. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python,” *Nature Methods*, vol. 17, pp. 261–272, 2020.
- [31] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, “API design for machine learning software: experiences from the scikit-learn project,” in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pp. 108–122, 2013.
- [32] T. Boggs, D. March, kormang, L. J. McGibbney, F. Magimel, G. Mason, K. Banman, M. J. L. Uzumaki), R. Kumar, T. G. Badger, T. Aarnio, W. Wang, and kidpixo, “spectralpython/spectral: Spectral python (spy) 0.23.1,” Oct. 2022.
- [33] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000.
- [34] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, “Transformers: State-of-the-art natural language processing,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, (Online), pp. 38–45, Association for Computational Linguistics, Oct. 2020.
- [35] R. Wightman, “Pytorch image models.” <https://github.com/rwightman/pytorch-image-models>, 2019.
- [36] Roboflow, “Supervision.”
- [37] C. M. Welsh, N. Fullard, C. J. Proctor, A. Martinez-Guimera, R. J. Isfort, C. C. Bascom, R. Tasseff, S. A. Przyborski, and D. P. Shanley, “Pycotools: a python toolbox for copasi,” *Bioinformatics*, vol. 34, no. 21, pp. 3702–3710, 2018.
- [38] S. Hoops, S. Sahle, R. Gauges, C. Lee, J. Pahle, N. Simus, M. Singhal, L. Xu, P. Mendes, and U. Kummer, “Copasi—a complex pathway simulator,” *Bioinformatics*, vol. 22, no. 24, pp. 3067–3074, 2006.

- [39] J. K. Medley, K. Choi, M. König, L. Smith, S. Gu, J. Hellerstein, S. C. Sealfon, and H. M. Sauro, “Tellurium notebooks—an environment for reproducible dynamical modeling in systems biology,” *PLOS Computational Biology*, vol. 14, no. 6, p. e1006220, 2018.
- [40] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, “DEAP: Evolutionary algorithms made easy,” *Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, jul 2012.
- [41] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, “Microsoft coco: Common objects in context,” 2015.
- [42] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, *et al.*, “Grounding dino: Marrying dino with grounded pre-training for open-set object detection,” *arXiv preprint arXiv:2303.05499*, 2023.
- [43] T. Ren, Q. Jiang, S. Liu, Z. Zeng, W. Liu, H. Gao, H. Huang, Z. Ma, X. Jiang, Y. Chen, Y. Xiong, H. Zhang, F. Li, P. Tang, K. Yu, and L. Zhang, “Grounding dino 1.5: Advance the .edge.of open-set object detection,” 2024.
- [44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” 2023.
- [45] R. Socher, M. Ganjoo, H. Sridhar, O. Bastani, C. D. Manning, and A. Y. Ng, “Zero-shot learning through cross-modal transfer,” 2013.
- [46] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, “Segment anything,” *arXiv:2304.02643*, 2023.
- [47] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer, “Sam 2: Segment anything in images and videos,” 2024.
- [48] D. Crockford, “The application/json media type for javascript object notation (json),” Tech. Rep. RFC 4627, Internet Engineering Task Force (IETF), 2006.
- [49] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, and H. Wu, “Mixed precision training,” 2018.

- [50] A. Agrawal, S. M. Mueller, B. M. Fleischer, X. Sun, N. Wang, J. Choi, and K. Gopalakrishnan, “Dlfloat: A 16-b floating point format designed for deep learning training and inference,” in *2019 IEEE 26th Symposium on Computer Arithmetic (ARITH)*, pp. 92–95, 2019.
- [51] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2015.
- [52] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.
- [53] G. R. Lee, R. Gommers, F. Wasilewski, K. Wohlfahrt, and A. O’Leary, “Pywavelets: A python package for wavelet analysis,” *Journal of Open Source Software*, vol. 4, no. 36, p. 1237, 2019.
- [54] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” 2018.
- [55] World Population Review, “Fig production by country 2025.” <https://worldpopulationreview.com/country-rankings/fig-production-by-country>, 2025. Retrieved September 19, 2025.
- [56] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, Z. Zeng, H. Zhang, F. Li, J. Yang, H. Li, Q. Jiang, and L. Zhang, “Grounded sam: Assembling open-world models for diverse visual tasks,” 2024.
- [57] Q. Jiang, F. Li, Z. Zeng, T. Ren, S. Liu, and L. Zhang, “T-rex2: Towards generic object detection via text-visual prompt synergy,” 2024.