FACULTAD DE ESTUDIOS ESTADÍSTICOS

MÁSTER EN MINERÍA DE DATOS E INTELIGENCIA DE NEGOCIOS

Curso 2024/2025

Trabajo de Fin de Máster

TÍTULO:

IDENTIFICACIÓN DE MONACITA GRIS EN TIEMPO REAL MEDIANTE UNA RED NEURONAL CONVOLUCIONAL

***Alumno*: MANUEL SEISDEDOS DOMÍNGUEZ**

***Tutora*: BELÉN RODRÍGUEZ CÁNOVAS**

Madrid, julio de 2025

UNIVERSIDAD COMPLUTENSE
MADRID

UNIVERSIDAD COMPLUTENSE MADRID

Declaración Responsable sobre Autoría y Uso Ético de Herramientas de Inteligencia Artificial (IA)

Yo, **SEISDEDOS DOMÍNGUEZ**

MANUEL

Con DNI/NIE/PASAPORTE:

declaro de manera responsable que el presente:

Trabajo de Fin de Máster (TFM)

Titulado

IDENTIFICACIÓN DE MONACITA GRIS EN TIEMPO REAL MEDIANTE UNA RED NEURONAL CONVOLUCIONAL

es el resultado de mi trabajo intelectual personal y creativo, y ha sido elaborado de acuerdo con los principios éticos y las normas de integridad vigentes en la comunidad académica y, más específicamente, en la Universidad Complutense de Madrid.

Soy, pues, autor del material aquí incluido y, cuando no ha sido así y he tomado el material de otra fuente, lo he citado o bien he declarado su procedencia de forma clara -incluidas, en su caso, herramientas de inteligencia artificial-. Las ideas y aportaciones principales incluidas en este trabajo, y que acreditan la adquisición de competencias, son mías y no proceden de otras fuentes o han sido reescritas usando material de otras fuentes.

Asimismo, aseguro que los datos y recursos utilizados son legítimos, verificables y han sido obtenidos de fuentes confiables y autorizadas. Además, he tomado medidas para garantizar la confidencialidad y privacidad de los datos utilizados, evitando cualquier tipo de sesgo o discriminación injusta en el tratamiento de la información.

En Madrid a 30 de junio de 2025

RESUMEN

El objetivo de este trabajo es el de definir un sistema de identificación y conteo de nódulos de mineral monacita gris mediante visión por ordenador. El interés al alza que tiene este mineral se debe a su alto contenido en óxidos de tierras raras, compuestos muy demandados en la industria tecnológica actual fundamentalmente por sus particulares características físico-químicas.

Para lograrlo se explorará la mejor configuración a la hora de entrenar una red neuronal convolucional para, a continuación, con los resultados de la exploración, entrenar y evaluar un modelo final a partir del cual se extraerán las conclusiones correspondientes.

Por sus buenos resultados al detectar objetos en tiempo real en una sólo pasada, se ha elegido una arquitectura YOLO (You Only Look Once) para desarrollar el proyecto.

Al no haberse localizado un conjunto de datos de acceso público adecuado para entrenar el modelo requerido por el objetivo enunciado, el TFM incluye la descripción detallada de la creación de un dataset de elaboración propia.

AGRADECIMIENTOS

Quiero agradecerle a Belén, mi tutora, su paciencia y apoyo. Igualmente, agradecer a Rafa Rincón y a Lucas Aguilar sus brillantes aportaciones; y, muy especialmente, a Enrique Burkhalter Thiébaut porque sin sus valiosas enseñanzas geológicas este trabajo no habría sido posible.

ÍNDICE GENERAL

ACRÓNIMOS

→ **AGPL:** GNU Affero General Public License, licencia de software libre que permite el uso, modificación y distribución de software, con la condición de que las modificaciones también sean libres.

→ **C2PSA**: Cross Stage Partial Spatial Attention, componente usado en YOLOv11 para mejorar la atención espacial en mapas de características optimizando la detección de objetos pequeños.

→ **CE**: Concesión de Explotación, concesión administrativa que identifica este derecho minero.

→ **CNN:** Convolutional Neural Network, red neuronal convolucional, usada principalmente en visión por ordenador para procesar imágenes.

→ **COCO:** Common Objects in Context, base de datos pública que contiene 330.000 imágenes clasificadas en 80 categorías.

→ **CV:** Computer Vision, visión por ordenador.

→ **DL:** Deep Learning, aprendizaje profundo.

→ **GPU:** Graphics Processing Unit, unidad de procesamiento gráfico, hardware especializado para acelerar cálculos en tareas como visión por ordenador.

→ **ILSVRC:** ImageNet Large Scale Visual Recognition Challenge, concurso anual basado en el dataset ImageNet para evaluar algoritmos de visión por ordenador.

→ **LSTM:** Long Short-Term Memory, es un tipo de red neuronal recurrente diseñada para modelar dependencias en datos secuenciales.

→ **LVIS:** Large Vocabulary Instance Segmentation, dataset de gran volumen para segmentación de instancias.

→ **MPV:** Mínimo Producto Viable.

→ **PE**: Permiso de Exploración, concesión administrativa que identifica este derecho minero.

→ **PI**: Permiso de Investigación, concesión administrativa que identifica este derecho minero.

→ **REE:** Rare Earth Elements, elementos de tierras raras.

→ **ReLU:** Rectified Linear Unit, función de activación usada en redes neuronales que introduce la no linealidad (f(x) = max(0,x)).

→ **REO:** Rare Earth Oxides, óxidos de tierras raras.

→ **SiLU:** Sigmoid Linear Unit, función de activación que combina una sigmoide con una transformación lineal.

→ **SPPF:** Spatial Pyramid Pooling Fusion, fusion de agrupación piramidal especial, componente dentro de la arquitectura de YOLOv11 cuyo objeto es mejorar la extracción de características a diferentes escalas.

→ **TPU:** Tensor Processing Unit, hardware especializado desarrollado por Google para acelerar cálculos dentro del DL.

→ **TTRR:** Tierras Raras.

→ **YOLO:** You Only Look Once, arquitectura de red neuronal convolucional para detección de objetos en tiempo real.

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ANEJOS

1. INTRODUCCIÓN

1.1 Las tierras raras

Los 17 elementos de la tabla periódica denominados tierras raras (TTRR), compuestos por los 15 elementos de la familia de los lantánidos añadiendo el escandio y el itrio, en las dos últimas décadas se han convertido en imprescindibles en la industria, especialmente en la industria tecnológica, y por ello objeto fundamental de la geopolítica mundial.

Prácticamente todas las TTRR tienen una aplicación industrial tecnológica debido a sus peculiares características físico-químicas. Actualmente los elementos de tierras raras más demandados en la industria son aquellos con propiedades magnéticas como pueden ser el neodimio, praseodimio, disprosio y el samario que aleado con cobalto preserva estas propiedades magnéticas aun a muy altas temperaturas. Son sólo unos ejemplos de los muchos que se podrían citar y que de manera exhaustiva quedan recogidos en la figura 1.

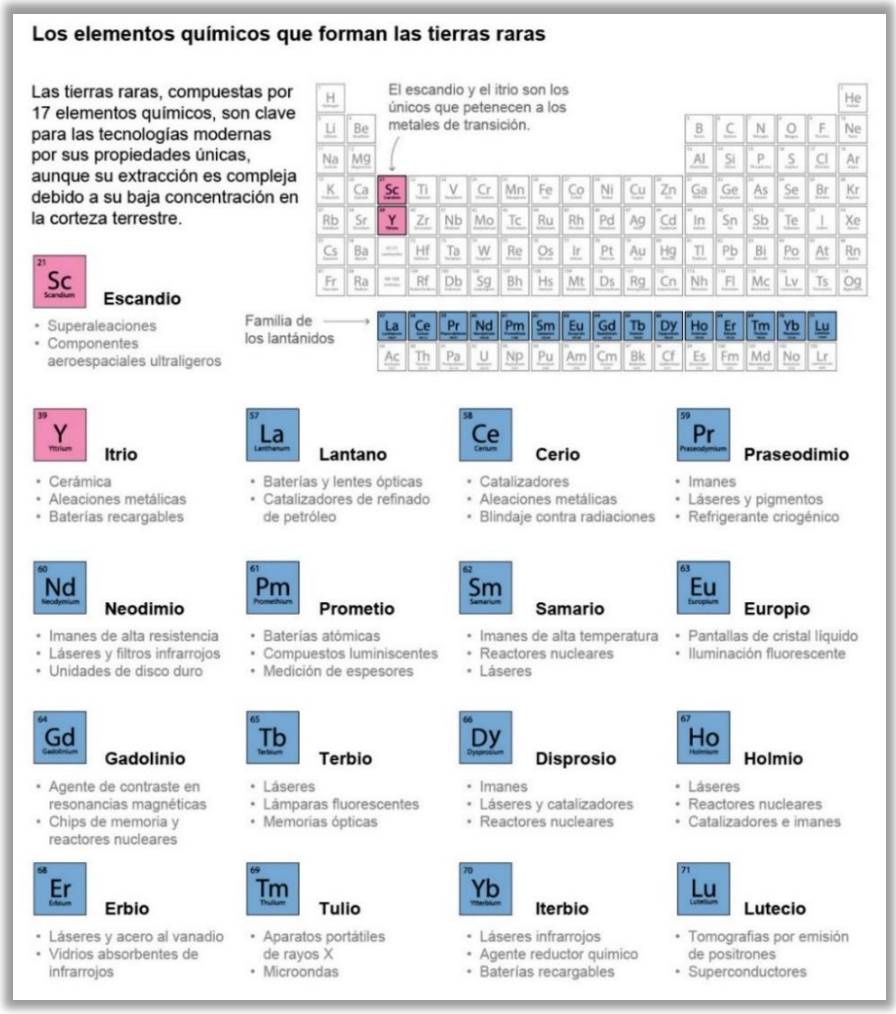


Figura 1: Las tierras raras en la tabla periódica detallando sus aplicaciones.

Fuente: Colegio Oficial de Geólogos, Mineral Commodity Summaries 2024, USGS y Statistical Review of World Energy.

Agencia EFE https://efs.efeservicios.com/noticia-multimedia/tierras-raras/55015920597

1.2 Hegemonía de China y consumo mundial de TTRR

Como se puede ver en la figura 2, en el año 2024, según el U.S. Geological Survey, la producción mundial de óxidos de tierras raras (REO) ha sido de alrededor 390.000 t y de ellas, cerca del 70% proceden de China. Las reservas de REO se encuentran por encima de los 90 Mt y alrededor del 50% se localizan en este país asiático.

	Mine production[e]		Reserves[11]
	2023	2024	
United States	41,600	45,000	1,900,000
Australia	[12]16,000	[12]13,000	[13]5,700,000
Brazil	140	20	21,000,000
Burma	[12]43,000	[12]31,000	NA
Canada	—	—	830,000
China	[14]255,000	[14]270,000	44,000,000
Greenland	—	—	1,500,000
India	2,900	2,900	6,900,000
Madagascar	[12]2,100	[12]2,000	NA
Malaysia	[12]310	[12]130	NA
Nigeria	[12]7,200	[12]13,000	NA
Russia	2,500	2,500	3,800,000
South Africa	—	—	860,000
Tanzania	—	—	890,000
Thailand	[12]3,600	[12]13,000	4,500
Vietnam	[12]300	[12]300	3,500,000
Other	1,440	1,100	NA
World total (rounded)	376,000	390,000	>90,000,000

Figura 2: Producción y reservas mundiales de TTRR.

Fuente: U.S. Geological Survey, Mineral Commodity Summaries, January 2025, pag. 145

China ha perfeccionado tecnologías para separar y purificar estos elementos logrando producir compuestos y metales de alta pureza necesarios para las aplicaciones industriales citadas. Más del 80% del refinado global de tierras raras se realiza en China con el consecuente control estratégico sobre la cadena de suministro.

No parece un tema coyuntural porque observando gráficos de las dos últimas décadas de consumos y precios se evidencian las tensiones ejercidas por la demanda y el suministro. Adamas Intelligence, consultora de reconocido prestigio en servicios estratégicos y minerales críticos, pronosticó en el año 2024 que el valor del consumo mundial de óxido de tierras raras magnéticas aumentará más de cinco veces para el año 2040, pasará de 7.800 millones de dólares en 2024 a 44.100 millones.

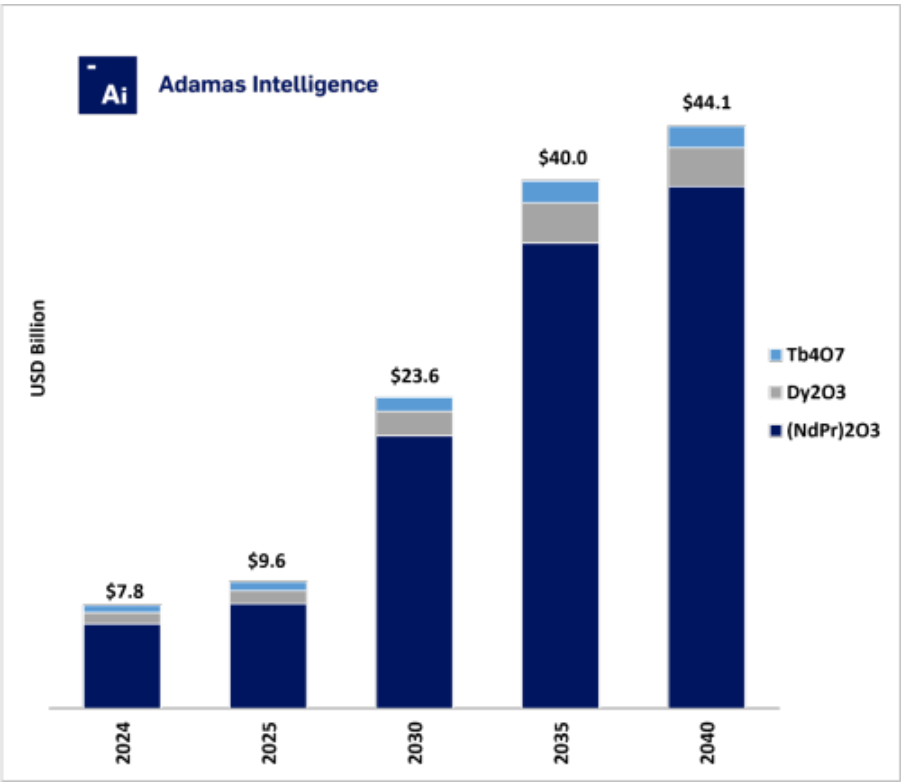


Figura 3: Demanda esperada para el consumo de REO magnéticos.
Fuente: Adamas Intelligence, tomada de https://www.greencarcongress.com/2024/09/20240902-adamas.html

1.3 Estrategia de la Unión Europea

Según *Herrera (2024),* la Unión Europea, teniendo en cuenta tres factores: la inestabilidad geopolítica, la creciente demanda de recursos y la concentración geográfica de su producción, se ha visto impulsada a desarrollar una política integral de materias primas que promueva el desarrollo de cadenas de suministro de materias primas seguras, responsables y sostenibles, como pilar fundamental para la economía europea para garantizar su sistema de bienestar.

Esta estrategia europea se inició en el año 2008 cuando se publicó la Iniciativa de Materias Primas de la UE donde ya se proponían medidas para reforzar la diplomacia de materias primas, aumentar su producción dentro de la UE y promover el reciclaje. En 2011 se creó la primera lista de materias primas críticas. Este listado, en el que están incluidas las TTRR, se ha ido actualizando cada tres años.

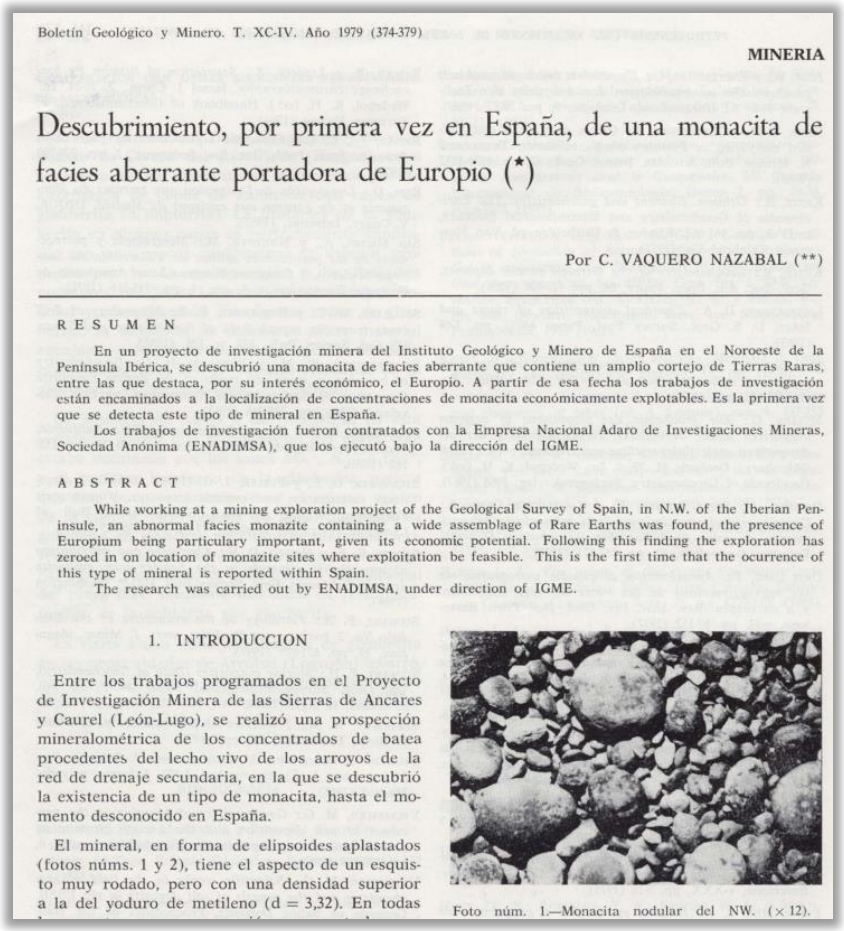
En el año 2024 se aprobó el Reglamento de Materias Primas Críticas de la UE, "***Reglamento (UE) 2024/1252 del Parlamento Europeo y del Consejo, de 11 de abril de 2024, por el que se establece un marco para garantizar un suministro seguro y sostenible de materias primas fundamentales"***, disposición legal cuya visión se centra en garantizar el suministro y reducir significativamente la dependencia exterior. Para ello, el reglamento fija unas prioridades de actuación y objetivos concretos definidos hasta el año 2030. Esta normativa apoya el desarrollo de proyectos de extracción, procesado y reciclaje en la UE. Además, exige que todos los Estados Miembros desarrollen e implementen, a través de sus Servicios Geológicos, planes nacionales de

exploración de materias primas críticas. Estos programas nacionales de exploración deben recopilar de manera sistemática datos geológicos, geofísicos y geoquímicos y analizarlos utilizando técnicas innovadoras que permitan descubrir depósitos minerales desconocidos, e impulsar la actividad de empresas de exploración minera junior.

1.4 La monacita gris

A las tierras raras, según *Regueiro (2019),* no se les denomina raras tanto por ser escasas sino, más bien, por una distribución natural desigual en la corteza terrestre; no es fácil encontrarlas en concentraciones suficientes para que resulte interesante su extracción desde un punto de vista económico. Además, estos elementos nunca aparecen aislados en la naturaleza, los podemos encontrar como óxidos metálicos contenidos en unos 25 minerales de los cuales los más importantes y que se explotan económicamente son: **monacita** (fosfato de tierras raras), **bastnasita** (flúor carbonato de tierras raras), **xenotima** (fosfato de itrio), **loparita** (óxido de cerio, sodio, calcio, titanio y niobio) y **gadolinita** (silicato de cerio, lantano, neodimio, itrio berilio y hierro).

Gil Ibarguchi (2025), cuenta cómo el ingeniero de minas Carlos Vaquero Nazabal, realizando trabajos de muestreo mediante batea en el sector central del batolito de Los Pedroches para su tesis doctoral a principios de los años 70 del pasado siglo, fue quien descubrió por primera vez en España la monacita nodular gris contenedora de TTRR.



Boletín Geológico y Minero. T. XC-IV. Año 1979 (374-379)

MINERIA

Descubrimiento, por primera vez en España, de una monacita de facies aberrante portadora de Europio (*)

Por C. VAQUERO NAZABAL (**)

RESUMEN

En un proyecto de investigación minera del Instituto Geológico y Minero de España en el Noroeste de la Península Ibérica, se descubrió una monacita de facies aberrante que contiene un amplio cortejo de Tierras Raras, entre las que destaca, por su interés económico, el Europio. A partir de esa fecha los trabajos de investigación están encaminados a la localización de concentraciones de monacita económicamente explotables. Es la primera vez que se detecta este tipo de mineral en España.

Los trabajos de investigación fueron contratados con la Empresa Nacional Adaro de Investigaciones Mineras, Sociedad Anónima (ENADIMSA), que los ejecutó bajo la dirección del IGME.

ABSTRACT

While working at a mining exploration project of the Geological Survey of Spain, in N.W. of the Iberian Peninsule, an abnormal facies monazite containing a wide assemblage of Rare Earths was found, the presence of Europium being particulary important, given its economic potential. Following this finding the exploration has zeroed in on location of monazite sites where exploitation be feasible. This is the first time that the ocurrence of this type of mineral is reported within Spain.

The research was carried out by ENADIMSA, under direction of IGME.

1. INTRODUCCION

Entre los trabajos programados en el Proyecto de Investigación Minera de las Sierras de Ancares y Caurel (León-Lugo), se realizó una prospección mineralométrica de los concentrados de batea procedentes del lecho vivo de los arroyos de la red de drenaje secundaria, en la que se descubrió la existencia de un tipo de monacita, hasta el momento desconocido en España.

El mineral, en forma de elipsoides aplastados (fotos núms. 1 y 2), tiene el aspecto de un esquisto muy rodado, pero con una densidad superior a la del yoduro de metileno (d = 3,32). En todas

Foto núm. 1.—Monacita nodular del NW. (× 12).

Figura 4: Extracto de un artículo de Carlos Vaquero en el Boletín Geológico y Minero, T. XC-IV.

Este mineral es una fuente primaria de óxidos de tierras raras (REO), pudiendo contener entre un 50% y un 70% en peso. Tiene forma de pequeños nódulos *(Vaquero, 1979)* milimétricos o submilimétricos de geometrías esféricas y elipsoidales. Comúnmente pueden presentar un ligero aplastamiento y coloración variable con tonos grises, amarillentos, rojizos o ligeramente verdosos. La superficie de los granos de monacita suele estar piqueteada y presenta un brillo mate en todas las variedades encontradas. La granulometría de los nódulos se encuentra entre los 0,1 y los 4,5 mm, concentrándose la mayor parte del mineral en la fracción comprendida entre los 0,5 y 1 mm. La densidad de la monacita, cuando la encontramos libre de esquistos, es de 4,65 kg/l. La susceptibilidad magnética es muy tenue.

Estas monacitas son de un tipo que se denominó “facies aberrante”, monacitas sedimentarias grises de origen diagenético; nada que ver con la monacita amarilla o endógena de origen ígneo.

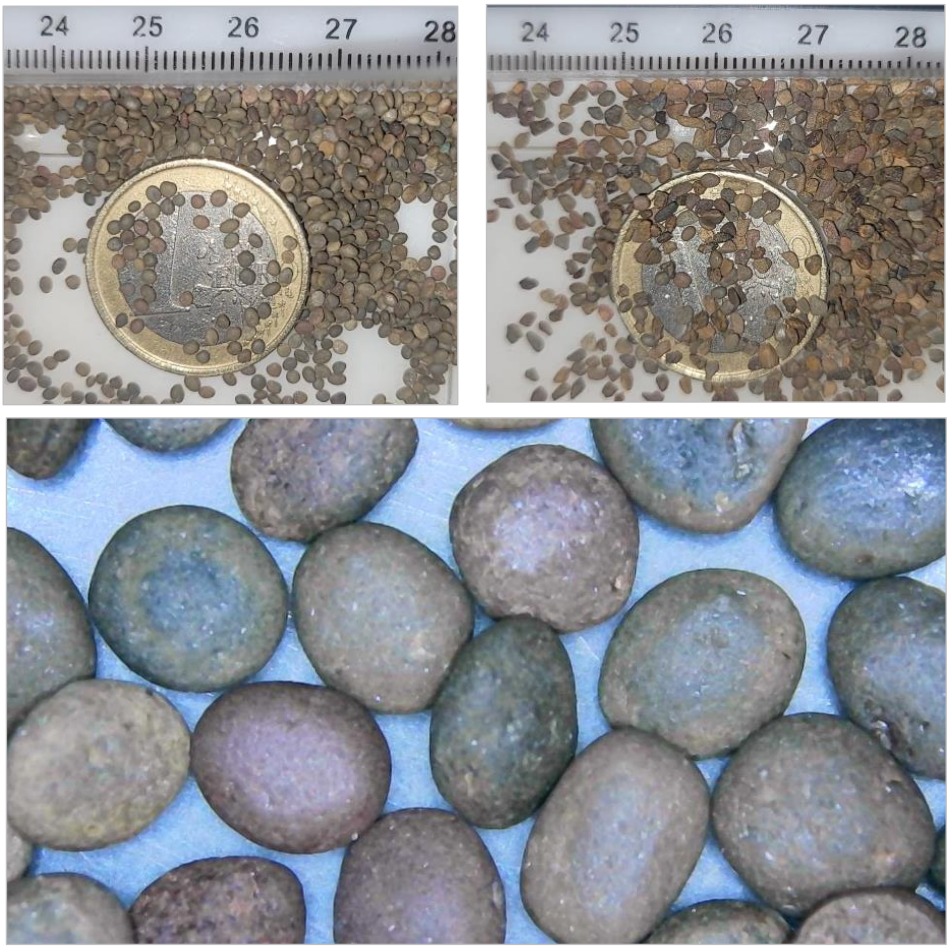


Figura 5: Muestras de monacita gris.

(Arriba a la izquierda, muestra de 100% monacita. A la derecha, concentrado con algún nódulo de monacita y mayoría de estéril. La foto inferior es una ampliación 50x de nódulos de monacita.)

Fuente: fotografías tomadas por el autor

Al hallazgo de Carlos Vaquero en el batolito de Los Pedroches le siguieron otros como el del área Ancares-Caurel, los Montes de Toledo o el Campo de Montiel. Ya en 1963 se habían identificado nódulos de monacita gris en la Bretaña francesa y más tarde apareció en concentrados de batea en Gales y el Pirineo francés (*Gil Ibarguchi, 2025*).

La teoría más aceptada, según *Donnot et al. (1973),* sobre el origen del mineral en estos hallazgos, plantea la formación de un gel de fosfato de tierras raras en limos marinos durante el Ordovícico Medio. Esta hipótesis explicaría su presencia en sedimentos recientes (placeres), que derivan esencialmente de la erosión de pizarras negras ordovícicas de *Calymene* ricas en materia orgánica. También pueden encontrarse los nódulos de monacita embebidos en la matriz pizarrosa encajante no alterada.

Otras ocurrencias de monacita gris, probablemente formados en otras edades geológicas y diferentes ambientes al descrito, se conocen en Siberia, Marruecos, Gabón, Congo, Madagascar, Canadá y EE. UU.

1.5 Exploración minera de la monacita gris

En la fase de exploración minera en general y referida a la exploración de monacita gris en particular, el geólogo, basándose en su conocimiento y después de realizar estudios previos en la fase de gabinete, estudios apoyados por bibliografía, antecedentes, mapas geológicos, topográficos, hidrogeológicos, etc., selecciona un área de interés con potencial para albergar un yacimiento de monacita.

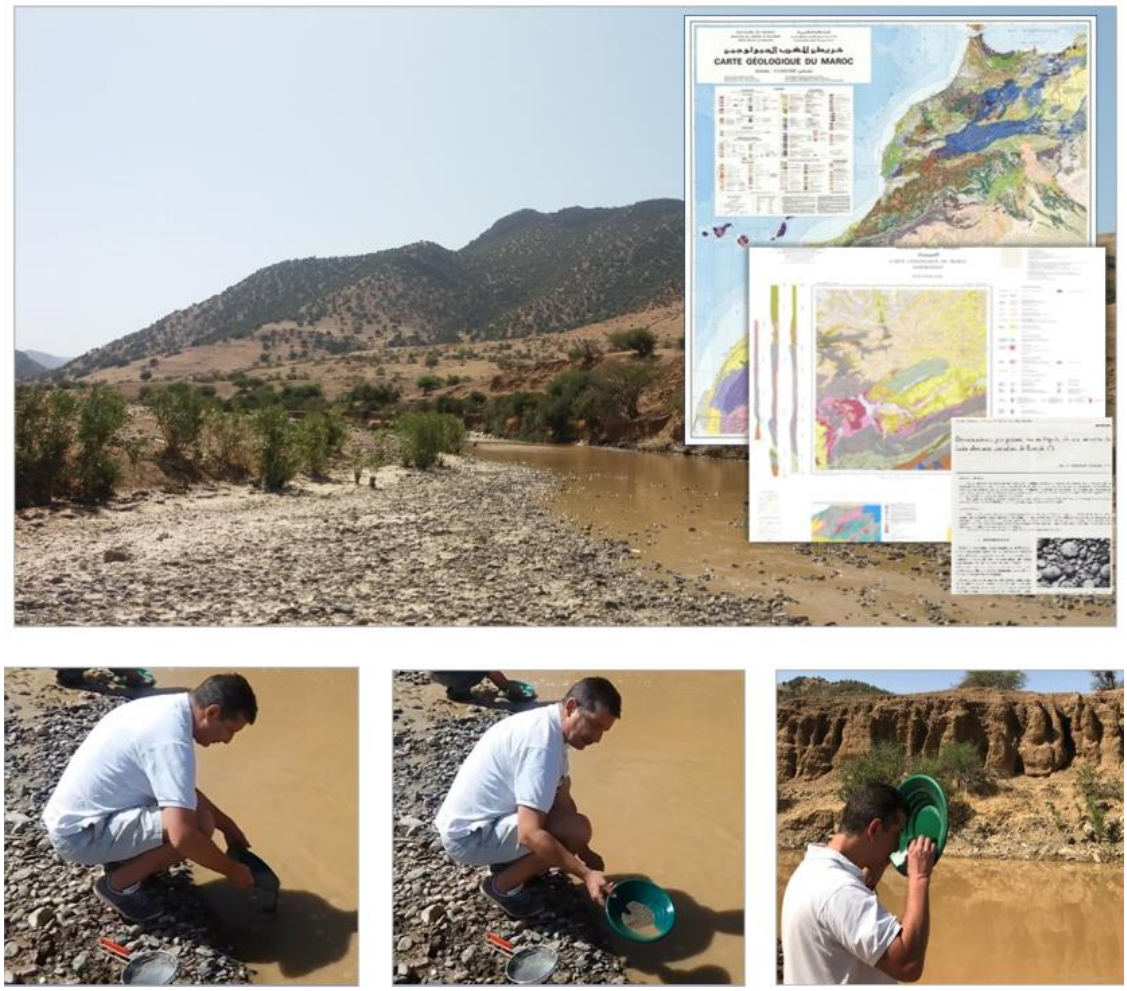


Figura 6: Exploración minera de monacita gris: el geólogo Enrique Burkhalter bateando una muestra de sedimento

Fuentes: fotografías tomadas por el autor. Mapas geológicos de www.sig-maroc.com

En este proceso, identifica ubicaciones interesantes para la toma de muestras, las cuales serán tratadas mediante bateo, técnica que aprovecha la alta densidad del mineral, en este caso 4,65 kg/l, para separarlo de los materiales más ligeros que componen la parte estéril. Una vez bateadas, las muestras se analizan visualmente para evaluar su contenido determinando la presencia y concentración del mineral objetivo.

1.6 Objetivos: Red Neuronal Convolucional para detectar nódulos de monacita gris

El presente TFM entrena, con un dataset propio construido con imágenes originales, una red neuronal convolucional (CNN) con el objetivo principal de conseguir un modelo lo suficientemente fiable y preciso que sea capaz de identificar y cuantificar, en tiempo real, sobre archivos de imágenes o de video, nódulos de mineral monacita gris cuando se encuentran liberados de su matriz encajante, acompañados, en mayor o menor cuantía, de material estéril, sobre muestras obtenidas en la fase de exploración geológica-minera.

Para la consecución del objetivo principal podemos identificar una serie de tareas que conforman los objetivos secundarios. Estos son:

- **Creación de un dataset**: construir un conjunto de datos compuesto por imágenes capturadas en condiciones controladas donde se identifiquen los nódulos de monacita gris anotándolos manualmente como la clase objetivo mediante cajas delimitadoras.

- **Entrenamiento del modelo**: entrenar un modelo de deep learning supervisado basado en una arquitectura de una CNN diseñado para detectar objetos en tiempo real.

- **Evaluación del rendimiento del modelo**: analizar el desempeño del modelo entrenado mediante métricas estándar.

1.7 Justificación

La técnica de Deep Learning y visión por ordenador consigue, a bajo coste, automatizar y estandarizar el proceso de identificar y cuantificar la cantidad de monacita contenida en el material concentrado durante la fase de exploración minera, mejorando notablemente la productividad.

El TFM se enmarca en un contexto geopolítico en el que las materias primas fundamentales, como las TTRR presentes en la monacita, han adquirido una importancia estratégica clave para la autonomía industrial y tecnológica de la UE. El desarrollo de tecnologías que abaraten la exploración y descubrimiento de estos materiales contribuye directamente a reducir la dependencia exterior y garantizar la estabilidad del suministro. En particular, este proyecto se alinea con los objetivos del ***Reglamento (UE) 2024/1252 del Parlamento Europeo y del Consejo, de 11 de abril de 2024, por el que se***

establece un marco para garantizar un suministro seguro y sostenible de materias primas fundamentales, promoviendo tanto la innovación tecnológica como la sostenibilidad en la cadena de valor de estos minerales críticos. Por todo ello, parece idóneo el tema elegido para tratar en este TFM que completa el Máster en Minería de Datos e Inteligencia de Negocios.

2. ESTADO DEL ARTE

2.1 Introducción

El presente estado del arte ofrece una revisión de los fundamentos, arquitecturas, hardware, herramientas, aplicaciones y tendencias actuales de las CNN, base del Deep Learning (DL) dentro del campo de la inteligencia artificial.

El DL ha transformado profundamente el campo de la inteligencia artificial al permitir que los sistemas desarrollen de manera autónoma representaciones complejas a partir de datos. Esta técnica utiliza redes neuronales profundas para modelar relaciones no lineales en datos. Las CNN destacan sobre todo en tareas de visión por ordenador, segmentación semántica y reconocimiento de voz. La calidad de las CNN sólo se explica por las mejoras significativas en hardware, el desarrollo de algoritmos más eficientes y la disponibilidad de conjuntos de datos de gran volumen que han conseguido aumentar la capacidad de los modelos a la hora de abordar problemas de alta complejidad.

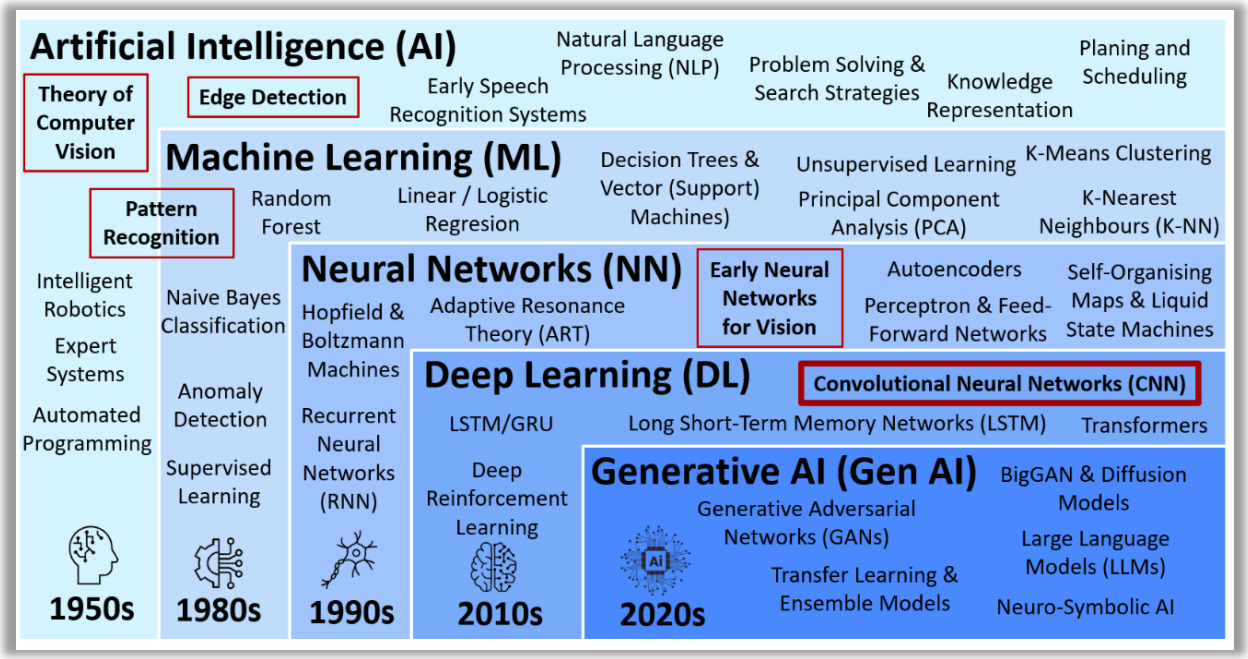


Figura 7: Redes Neuronales Convolucionales contextualizadas cronológicamente.
Fuente: infografía elaborada por el autor a partir de la integración de distintas fuentes bibliográficas*.

2.2 Evolución histórica contextualizada de las CNN

Desde el año 1943, cuando Warren McCulloch y Walter Pitts presentaron el primer modelo de la neurona artificial, una idea innovadora que buscaba imitar el funcionamiento de las neuronas humanas y que sentaron las bases de la inteligencia artificial, hasta las actuales IAs generativas multimodales, han sido muchos los hitos que se pueden destacar en cuanto a las CNN pero, probablemente, muchos menos de los que se han quedado por el camino; intentos fallidos de arquitecturas y tecnologías que fueron superadas por los que en este TFM se citan.

Se puede registrar como primer hito importante con entidad propia en cuanto a las CNN el Neocognitron de Kunihiko Fukushima en 1979, un antecesor de las redes neuronales convolucionales que introdujo la capacidad de procesar patrones visuales de manera organizada y jerárquica. Casi una década después, en 1988, nos encontramos con LeNet-5 que fue la primera CNN funcional para el reconocimiento de dígitos escritos a mano que demostró gran eficacia, por lo que abrió la puerta a aplicaciones prácticas.

En 2012, AlexNet, aprovechando los avances en hardware, los frameworks que resolvían la computación distribuida y los datasets de grandes volúmenes, supuso una revolución al combinar enormes cantidades de datos con la potencia de las unidades de procesamiento gráfico (GPU), logrando un rendimiento espectacular en la clasificación de imágenes.

El progreso en redes neuronales no puede entenderse sin el papel determinante del hardware. La adopción generalizada de GPUs, y más adelante de TPUs, ha permitido reducir de manera importante los tiempos de entrenamiento, pasando de días a horas, lo que ha facilitado la iteración de modelos modificando hiperparámetros. Herramientas como CUDA y librerías como cuDNN, junto con frameworks de alto nivel como PyTorch, han contribuido enormemente a la generalización en los desarrollos de modelos complejos.

Además, el avance ha estado impulsado por la existencia de benchmarks públicos y competiciones internacionales que han establecido estándares de comparación entre modelos. Concursos como ImageNet, COCO o PASCAL VOC al proporcionar conjuntos de datos bien definidos y métricas como el mean Average Precision (mAP), han fomentado una cultura de mejora iterativa y validación objetiva sirviendo para discernir y cribar aquellas innovaciones con recorrido frente a las que se han ido quedando por el camino. Igualmente, estos entornos han permitido identificar claramente qué aspectos mejorar en cada nueva arquitectura, acelerando el perfeccionamiento de los modelos convolucionales.

Sólo así se entiende la aceleración de las CNN a partir del año 2012. En la tabla 1, se resume cronológicamente los principales hitos en la historia de las arquitecturas de redes neuronales poniendo énfasis ***(en negrita) en aquellas que han influido directamente en el desarrollo y perfeccionamiento de las CNN***. Se incluyen no solo avances teóricos y arquitectónicos, sino también modelos aplicados, algoritmos de entrenamiento, enfoques híbridos y los *hitos que han marcado los avances en el hardware (en tono granate)* junto con los *principales concursos y conjuntos de datos (identificados en color azul).* Esta integración ordenada cronológicamente resulta de gran ayuda para entender la evolución de las CNN.

Se han incluido hitos de la familia de modelos YOLO (You Only Look Once), reconocida por su eficiencia en tareas de detección de objetos. YOLOv11, el modelo utilizado en este TFM, representa la penúltima evolución de la línea (durante la redacción de este trabajo se ha publicado la versión 12 de la serie) que introduce mejoras significativas en precisión y eficiencia de cómputo mediante el refinamiento de componentes convolucionales clásicos sin haber recurrido a arquitecturas basadas en

Transformers. Su rendimiento ha sido validado en benchmarks como COCO, LVIS y OBB-DOTA v1, posicionándolo como una solución de referencia para aplicaciones en tiempo real con recursos limitados. En este año 2025 se cumple la década de la primera arquitectura YOLO por lo que parece una tecnología de largo recorrido a tener en cuenta.

Año	Modelo / Arquitectura	Descripción	Categoría	Autor/es
1943	Neurona de McCulloch-Pitts	Primer modelo lógico formal de neurona.	Fundamento teórico	Warren McCulloch, Walter Pitts
1958	Perceptrón	Primer modelo entrenable para clasificación lineal.	Arquitectura	Frank Rosenblatt
1974	Multilayer Perceptron (MLP)	Introduce redes multicapa, base para retropropagación (se aplicó a partir de 1986)	Arquitectura	Paul Werbos
1979	**Neocognitron**	**Inspiración para CNN con jerarquías de características.**	**Arquitectura (precursora CNN)**	**Kunihiko Fukushima**
1986	Backpropagation + sigmoides	Algoritmo clave para entrenar redes profundas.	Algoritmo / Optimización	David Rumelhart, Geoffrey Hinton, Ronald Williams
1989	Elman RNN	Introduce memoria dinámica para secuencias.	Arquitectura (RNN)	Jeffrey Elman
1988-1998	**LeNet-5**	**Primera CNN funcional (1998) para reconocimiento de dígitos.**	**Arquitectura (CNN)**	**Yann LeCun, Léon Bottou, Yoshua Bengio y Patrick Haffner**
1997	LSTM	Mitiga el desvanecimiento del gradiente en secuencias largas.	Arquitectura (RNN)	Sepp Hochreiter, Jürgen Schmidhuber
2006	DBN + Autoencoders	Inicio del Deep Learning moderno con aprendizaje no supervisado.	Arquitectura no supervisada	Geoffrey Hinton, Neal Smolensky
2006	**CUDA**	**Framework de NVIDIA para cálculos paralelos en GPUs, clave para CNN.**	**Hardware (plataforma)**	**NVIDIA**
2009	ImageNet	Conjunto de datos masivo de imágenes etiquetadas, base para ILSVRC y el auge del aprendizaje profundo.	Dataset / Benchmark	Fei-Fei Li, Jia Deng, et al.
2010	PASCAL VOC	Dataset con anotaciones detalladas y métrica mAP; referencia clave para tareas de detección de objetos.	Dataset / Benchmark	Mark Everingham et al.
2010	Kaggle	Plataforma de competiciones y datasets para ciencia de datos y aprendizaje automático, impulsa avances en CNNs, Transformers, y modelos como YOLO.	Concurso/Plataforma	Anthony Goldbloom
2010-2017	ILSVRC	Concurso basado en ImageNet, cataliza el auge de CNNs con AlexNet.	Concurso / Benchmark	Fei-Fei Li, Jia Deng, et al. (ImageNet)

Año	Modelo / Arquitectura	Descripción	Categoría	Autor/es
2012	AlexNet	Revolución en CNNs: GPUs, ReLU, triunfo en ImageNet.	Arquitectura (CNN)	Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton
2012	NVIDIA GTX 580 / Kepler	GPUs accesibles que aceleraron AlexNet, popularizando Deep Learning.	Hardware	NVIDIA
2014	ZFNet	Optimiza AlexNet y visualiza filtros para interpretabilidad.	Optimización (CNN)	Matthew Zeiler, Rob Fergus
2014	VGGNet	CNN profunda con capas uniformes y bloques simples.	Arquitectura (CNN)	Karen Simonyan, Andrew Zisserman
2014	GoogLeNet / Inception	Arquitectura eficiente con módulos Inception.	Arquitectura (CNN)	Christian Szegedy et al.
2014	COCO	Dataset para detección de objetos y segmentación, estándar para YOLO y YOLOv11.	Dataset / Benchmark	Lin, TY. et al
2015	ResNet	Conexiones residuales para redes muy profundas.	Arquitectura (CNN)	Kaiming He et al.
2015	YOLO	Detección de objetos en tiempo real con CNN.	Aplicación (visión por ordenador)	Joseph Redmon et al.
2016	DenseNet	Conexiones densas para mejorar propagación del gradiente.	Arquitectura (CNN)	Gao Huang et al.
2016	WaveNet	Modelo generativo para audio basado en CNN dilatadas.	Arquitectura (CNN generativa)	Aaron van den Oord et al.
2016	TensorFlow	Framework de Google con el API Keras, simplifica desarrollo de CNN.	Herramienta	Google Brain
2016	Google TPU v1	Primer ASIC para Deep Learning, optimizado para TensorFlow.	Hardware	Google
2017	MobileNet / SqueezeNet	CNNs ligeras optimizadas para dispositivos móviles.	Arquitectura (CNN eficiente)	Andrew Howard et al., Forrest Iandola et al.
2017	NASNet	CNN optimizada mediante búsqueda automática de arquitectura.	Meta-arquitectura (NAS)	Barret Zoph, Quoc Le
2017	Transformer	Modelo basado en atención para procesamiento de secuencias.	Arquitectura (Transformers)	Ashish Vaswani et al.
2017	PyTorch	Framework flexible, líder en investigación de CNN y YOLOv11.	Herramienta	Facebook AI Research (FAIR)

Año	Modelo / Arquitectura	Descripción	Categoría	Autor/es
2017	NVIDIA V100	GPU con núcleos tensoriales, acelera modelos como BERT.	Hardware	NVIDIA
2017	Apple Neural Engine	Chip para inferencia de CNN en móviles, soporta aplicaciones en tiempo real.	Hardware (Edge)	Apple
2017	Google TPUs	TPUs optimizadas para TensorFlow, con alta eficiencia energética.	Hardware	Google
2018	BERT	Transformer bidireccional para tareas de NLP.	Arquitectura (Transformers)	Jacob Devlin et al.
2018	OBB-DOTA	Dataset para detección de objetos orientados en imágenes aéreas, evalúa YOLOv11.	Dataset / Benchmark	Jian Ding et al.
2019	EfficientNet	CNN escalada automáticamente para eficiencia y precisión.	Arquitectura (CNN eficiente)	Mingxing Tan, Quoc Le
2019	NVIDIA A100	GPU con precisión mixta, ideal para entrenar YOLOv11 en PyTorch.	Hardware	NVIDIA
2019	LVIS	Dataset para segmentación de instancias con vocabulario grande, evalúa YOLOv11.	Dataset / Benchmark	Facebook AI Research
2020	DETR	Detección de objetos con Transformers y CNN.	Arquitectura híbrida	Nicolas Carion et al.
2020	Vision Transformer (ViT)	Transformer puro aplicado a tareas de visión.	Cambio de paradigma	Alexey Dosovitskiy et al.
2020	Ultralytics YOLO	Biblioteca PyTorch para YOLO, optimiza detección de objetos.	Herramienta	Ultralytics
2021	ConvNeXt, CoAtNet, Swin Transformer	Modelos híbridos que combinan CNN y atención.	Arquitectura híbrida	Zhuang Liu et al., Google Brain
2021	DALL·E, CLIP	Modelos multimodales para generación y clasificación texto-imagen.	Multimodalidad	OpenAI
2021	Jetson Orin / Coral TPU	Chips para inferencia en IoT, compatibles con YOLOv11 en tiempo real.	Hardware (Edge)	NVIDIA, Google
2022	Stable Diffusion	Modelo de difusión abierto para generación de imágenes.	Difusión / Generación	Jonathan Ho et al., CompVis
2023	Segment Anything (SAM)	Segmentación universal con prompts interactivos.	Aplicación (visión generalista)	Meta AI
2023	GPT-4	Modelo multimodal para texto e imágenes.	Fundación / Multimodalidad	OpenAI

Año	Modelo / Arquitectura	Descripción	Categoría	Autor/es
2024	***YOLOv11***	***Detección de objetos en tiempo real con mayor precisión y menos parámetros.***	***Aplicación (visión por ordenador)***	***Ultralytics***
2024	*GPT-4V / Gemini / Claude 3*	*Modelos multimodales con capacidades visuales generalistas.*	*Multimodalidad*	*OpenAI, Google DeepMind, Anthropic*
2024	*Florence-2*	*Modelo multimodal para tareas visuales y lingüísticas.*	*Multimodalidad*	*Microsoft Research*
2024	***Cerebras Wafer-Scale Engine***	***Chip de escala masiva para entrenamiento de modelos profundos.***	***Hardware***	***Cerebras Systems***
Tendencia para 2025	*Modelos híbridos CNN + Transformer + Razonamiento*	*Convergencia de CNN, Transformers y razonamiento para AGI visual.*	*Predicción / Hibridación*	*Comunidad internacional*

Tabla 1: Cronología contextualizada de hitos clave en la evolución de las Redes Neuronales Convolucionales.

Fuente: tabla elaborada por el autor a partir de la integración de distintas fuentes bibliográfica**.

2.3 Aplicaciones de visión por ordenador en geociencias y mineralogía

En campos como las geociencias o la mineralogía, la clasificación de rocas y minerales es un proceso necesario tanto con fines económicos e industriales como por puro interés científico. Esta identificación de rocas y minerales históricamente se ha basado en la interpretación visual realizada por profesionales especializados. La identificación y descripción de minerales, ya sea a partir de muestras en bruto o de secciones laminares de minerales, suelen ser tareas exclusivas de geólogos que analizan colores, tonalidades, texturas, patrones microscópicos, etc.

Inicialmente se emplearon técnicas de reconocimiento de patrones visuales sobre imágenes de microscopía electrónica con el fin de entrenar modelos capaces de identificar diferentes tipos de minerales y distinguir estructuras internas de los mismos como inclusiones o fracturas. Pero pronto proliferaron los casos de uso referentes a las ciencias de la tierra explorando todas las escalas posibles, desde el uso de CNN para la detección de estructuras naturales a partir de imágenes aéreas o la clasificación de minerales en mano hasta la ya citada identificación de patrones microscópicos a partir de láminas delgadas.

La ventaja no es sólo la precisión y rapidez de las clasificaciones sino, más bien, los beneficios que reporta poder diseñar procesos automáticos que industrialicen determinados procedimientos entre los que se encuentran las prospecciones mineras. Igualmente resulta interesante poder crear conjuntos de datos personalizados utilizando imágenes recopiladas por el propio investigador, como es el caso del presente TFM, en situaciones donde los datos públicos son escasos o poco representativos.

3. REDES NEURONALES CONVOLUCIONALES

3.1 Fundamentos de las CNN

Como se ha visto en el apartado 2, las CNN representan un subconjunto dentro del DL que destacan por su capacidad para extraer características de manera automática y jerárquica a partir de datos de entrada formados por imágenes.

Aplicadas a la detección de objetos, estas redes, están inspiradas en el funcionamiento del sistema visual humano. Utilizan capas convolucionales, funciones de activación, que introducen la no linealidad para permitir a la red aprender patrones más complejos que van más allá de las combinaciones lineales simples como *ReLU* o *Softmax*, y operaciones de agrupamiento *pooling* para procesar los datos estructurados de las imágenes.

Las capas convolucionales aplican filtros que identifican patrones locales o de alto nivel como pueden ser los bordes y las texturas, mientras que las capas de *pooling* reducen la dimensionalidad persistiendo características relevantes. Este diseño permite a las CNN permanecer invariantes a transformaciones como traslaciones o rotaciones, lo que las hace especialmente efectivas en tareas de visión por ordenador.

Las capas totalmente conectadas, que suelen encontrarse cerca del final de la red, conectan todas las neuronas de la capa anterior con todas las neuronas de la capa actual. Utilizan las características de alto nivel extraídas por las capas convolucionales y de agrupación para realizar tareas de clasificación o regresión, como asignar una etiqueta final a la imagen.

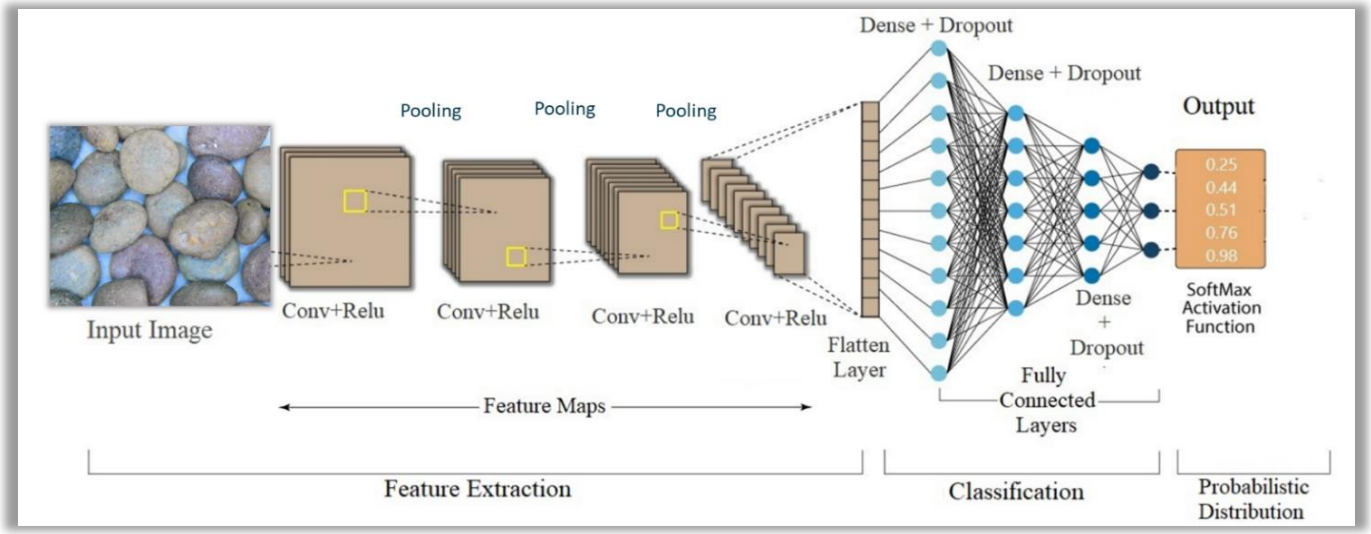


Figura 8: Arquitectura clásica de una red neuronal convolucional.

Fuente: figura modificada por el autor a partir de Junayed, Masum Shah et al.

Resumiendo, las CNN se diferencian de otras redes por su capacidad de procesar datos espaciales, como son las imágenes, aplicando una jerarquía espacial que permite detectar desde patrones simples dentro de la imagen, hasta el reconocimiento de las formas de los objetos. Su eficiencia se debe, en parte, a su capacidad de compartir parámetros aplicando un mismo filtro a distintas áreas de la imagen. Esto evita el sobreajuste. Igualmente, estas arquitecturas, consiguen reconocer objetos en diversas posiciones por su capacidad de permanecer invariantes a la traslación.

3.2 Fundamentos de la arquitectura de YOLOv11

En el presente TFM se ha trabajado con una arquitectura YOLOv11 desarrollada por Ultralytics. Esta arquitectura emplea redes neuronales convolucionales optimizadas para realizar detección de objetos en tiempo real. YOLOv11 se distribuye bajo licencia AGPL-3.0 (Affero General Public License v3.0), lo que implica la obligación de liberar las modificaciones del software si se utiliza en servicios accesibles por red. En este trabajo se ha empleado sin modificación directa del código base, respetando los términos de la licencia y con fines exclusivamente académicos y no comerciales.

El repositorio oficial del proyecto se encuentra disponible en:
https://github.com/ultralytics/ultralytics

YOLO es una familia de modelos de detección de objetos en tiempo real que combina precisión y velocidad. Tecnología adecuada para aplicaciones de videovigilancia, conducción autónoma, análisis de imágenes médicas, gestión de inventarios, identificación de minerales, monitoreo de cultivos, etc. YOLO realiza la detección en una sola pasada, prediciendo simultáneamente cuadros delimitadores (bounding boxes), las clases de objetos y el índice de confianza de cada predicción.

En relación a la arquitectura de YOLOv11, se puede decir que cuenta con tres componentes clave representados en la figura 9: backbone, neck y head. Estos componentes, en conjunto, conforman un proceso integral para la detección de objetos, que abarca desde el preprocesamiento de las entradas hasta la generación de la salida final. Inicialmente, la imagen de entrada sin procesar se somete a etapas de preprocesamiento como la normalización, el redimensionamiento y a menudo se emplean técnicas de aumento de datos (data augmentation), como rotaciones, cambios de brillo o mosaicos, para así mejorar la robustez del modelo. Posteriormente, la imagen preprocesada se introduce en el módulo principal de extracción de características backbone. Estos mapas de características, que contienen información local y global de gran valor, se trasladan posteriormente a neck para una integración avanzada.

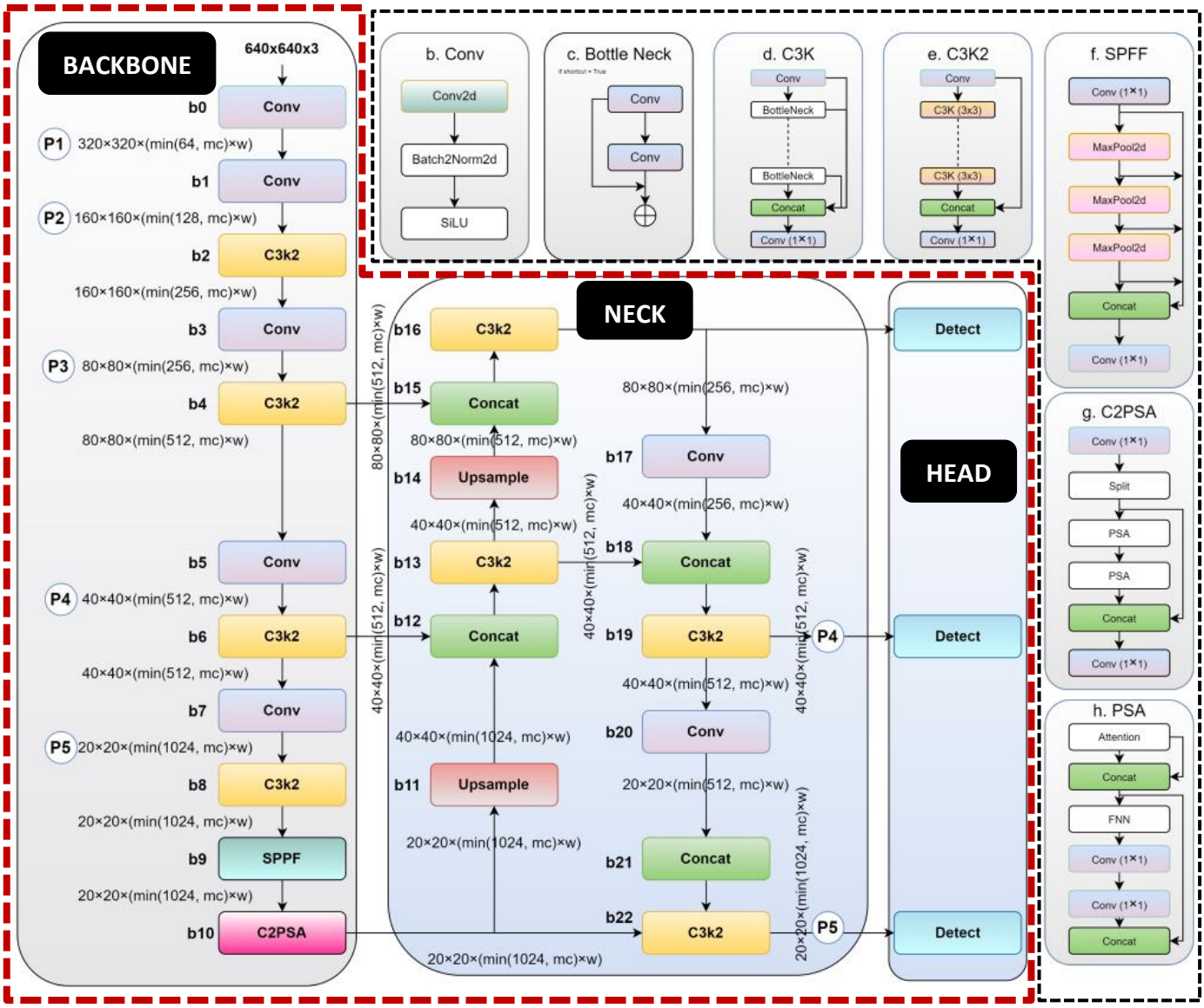


Figura 9: Arquitectura general de YOLOv11

Fuente: figura modificada por el autor a partir de Rasheed, A. F et al (2024).

Backbone: es una CNN que procesa la imagen de entrada para generar mapas de características, esto es, representaciones jerárquicas de características visuales como bordes, texturas y patrones de alto nivel. Actúa como el generador de características de las imágenes capturando información esencial que el resto de la red utiliza para las tareas de detección, identificación y localización de objetos. Las capas iniciales buscan patrones simples y las capas más profundas capturan características complejas y abstractas como pueden ser las formas de los objetos. Esta red ha sido preentrenada con ImageNet (ver tabla 1) para tareas de clasificación de imágenes por lo que la red se inicializa con pesos que pueden reconocer características visuales generales.

Neck: parte fundamental de la arquitectura ya que es donde se agregan y refinan las características extraídas por backbone antes de pasarlas a head donde se resolverán las predicciones finales. Utilizando estructuras que mejoran la detección de objetos de distintos tamaños, se fusionan características de diferentes escalas para mejorar la

capacidad que tiene el modelo de detectar objetos de diferentes dimensiones, resoluciones y orientaciones.

Estos dos componentes, backbone y neck, incluyen diversos módulos como son las capas convolucionales **Conv**, módulos **C3k2**, módulos **Spatial Pyramid Pooling-Fast (SPPF)** y **Cross Stage Partial Spatial Attention (C2PSA)** detallados en la figura 9.

El módulo **Conv** gestiona el procesamiento inicial de características, mientras que el módulo **C3k2** emplea una arquitectura de múltiples ramas para extraer tanto detalles de grano fino como información jerárquica profunda. Por otro lado, el módulo **SPPF** integra información contextual multiescala mediante agrupación piramidal. El módulo **C2PSA** mejora aún más la representación de características al aprovechar la atención del espacio de píxeles a través de escalas cruzadas, lo que aumenta la capacidad del modelo para centrarse en regiones objetivo críticas.

Adicionalmente, aparecen dos módulos más; el módulo **Concat** donde se combinan, se concatenan, las salidas de dos o más capas o ramas del modelo a lo largo de un eje con el objeto de fusionar características de diferentes niveles o resoluciones, y el módulo **Upsample** que permite la operación de aumentar la resolución espacial de una imagen o mapa de características, normalmente duplicando o interpolando los valores de los píxeles para que el tamaño de salida sea mayor que el de entrada.

Las funciones de activación son fundamentales al introducir no linealidades que permiten al modelo aprender patrones complejos. En las capas convolucionales del backbone y el neck se utiliza la función **SiLU** (Sigmoid Linear Unit), función que mejora la convergencia y la precisión en comparación con ReLU, especialmente en tareas de detección de objetos en tiempo real.

Head: es el componente encargado de predecir los cuadros delimitadores (bounding boxes) y las probabilidades de las clases objetivo. En comparación con versiones anteriores, YOLOv11 consigue que este módulo sea más ligero y preciso, optimizado mediante técnicas avanzadas como la regresión sin anclajes. En este componente se utilizan funciones de activación como **SiLU** en las capas convolucionales y **sigmoide** para las probabilidades de clase **CLSLoss**, mientras que las predicciones de localización **CIoU** ajustan coordenadas con **transformaciones logísticas**. Las pérdidas CLSLoss y CIoU optimizan la clasificación y localización, respectivamente, con un diseño eficiente que incluye convoluciones de profundidad separable **DWCov**.

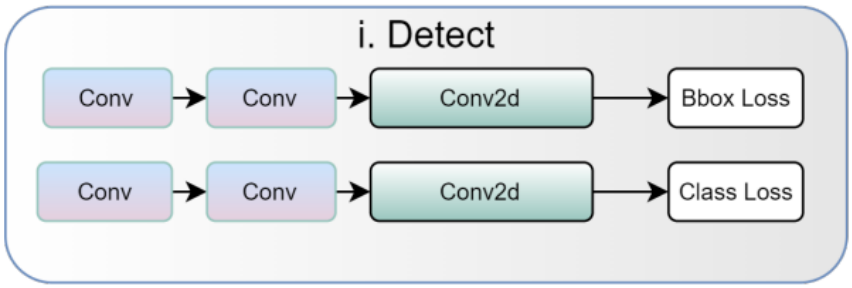


Figura 10: Detalle del módulo detect en head

Fuente: Rasheed, A. F et al (2024).

4. CONSTRUCCIÓN DEL DATASET

No se ha encontrado un conjunto de datos de acceso público específico de monacita gris que pueda ser usado en este TFM por lo que, en el presente apartado, se describe cómo se ha construido el dataset que entrenará el modelo.

4.1 Calidad del conjunto de datos

Respecto a las directrices de calidad a la hora de constituir el conjunto de datos para visión por ordenador, se ha atendido a la documentación *Ultralytics (2023)* donde se destacan unos puntos esenciales que a continuación se comentan en relación al caso de uso del TFM:

- **Precisión:** los datos deben reflejar fielmente situaciones reales y estar correctamente etiquetados. En el caso de la monacita, cada nódulo debe ser etiquetado con precisión tanto geométrica como cualitativa.

- **Diversidad:** un buen conjunto de datos incluye diversos ejemplos para que el modelo funcione correctamente en diferentes situaciones. En el caso particular del TFM deben registrarse la mayor variedad de muestras en cuanto a concentraciones de monacita, iluminaciones, tonalidades, ampliaciones ópticas, etc.

- **Consistencia:** los datos de alta calidad deben seguir un formato estándar de calidad uniforme. Para la identificación de monacita, las imágenes del dataset, deben de tener resoluciones óptimas para que el modelo pueda aprender las características del mineral en diferentes situaciones.

- **Actualización temporal:** una característica de la calidad de los conjuntos de datos es la adaptación de los registros a nuevas realidades. Como hemos visto, los nódulos de monacita gris, a pesar de su nombre, tienen variedad de colores, pero podría darse el caso de yacimientos de monacita con un sesgo de color característico determinado por el sedimento en el que se encuentran; terrenos muy arcillosos devolverán unos nódulos predominantemente rojizos, así como sedimentos cementosos devolverán minerales más grisáceos. En este caso conviene actualizar el dataset reflejando ese sesgo.

4.2 Tamaño del conjunto de datos

Igualmente, respecto al tamaño del dataset atendiendo a *Ultralytics, (s.f.)* para asegurar la calidad y consistencia del conjunto de datos, documentación donde se indica que, sin entrar a valorar la complejidad de los objetos y las particularidades de cada caso, como norma general, para un entrenamiento por trasferencia de un dataset de una sola clase y para asegurarse una detección de objetos eficaz, se disponga de un conjunto como mínimo de 100 imágenes.

Por otro lado, *Everingham et al. (2010),* donde se describe el dataset/benchmark PASCAL VOC (tabla 1), se recoge que en los datasets clásicos se lograron buenos resultados entrenando CNN con conjuntos de datos de entre 500-2.000 imágenes por clase. Además se destacan los problemas de performace a causa de errores en las anotaciones.

El conjunto de datos COCO (Common Objects in Context), descrito por *Lin et al. (2014)*, dataset de amplia difusión y que contiene 330.000 imágenes de las que 200.000 tienen anotaciones distribuidas en 80 categorías de objetos, fue creado para superar las limitaciones de los conjuntos anteriores como PASCAL VOC; a mayor volumen de datos mejor generalización, menor riesgo de overfitting, mayor diversidad de ejemplos y mejores rendimientos, como se ha demostrado empíricamente en concursos como ILSVRC (Imagenet Large Scale Visual Recognition Challenge).

Con esta información se pueden identificar tres tipos de dataset en cuanto al volumen de imágenes, siempre teniendo en cuenta que son valores orientativos y que el rendimiento final del modelo depende de múltiples variables además de la cantidad de imágenes registradas en el conjunto de datos. En la siguiente tabla se recoge la propuesta comentada:

Tipo de aplicación	Tamaño (nº de imágenes)	Justifica
Fase experimental	100-200	*Ultralytics, (s.f.)*
Modelo funcional	500-2.000	*Everingham et al. (2010)*
Producción	> 2.000	*Lin et al. (2014)*

Tabla 2: Propuesta de categorización de datasets por el volumen de imágenes

Fuente: tabla elaborada por el autor

De la misma manera, hay que resaltar que estamos ante un caso de uso un tanto particular donde en cada imagen puede haber numerosas instancias (bounding boxes) de la misma clase. De tal manera que, si tenemos 100 imágenes con una media aritmética de 10 instancias/imagen, en realidad estamos poniendo a disposición del modelo 1.000 representaciones de nódulos de monacita. No obstante, estas 1.000 instancias, no enriquecerán de modo equivalente un modelo a como lo harían 1.000 imágenes distintas puesto que las instancias pertenecientes a una misma imagen compartirán numerosas características como pueden ser el zoom óptico, la resolución de la imagen, la iluminación del entorno o la concentración de mineral.

En este TFM, parece lógico plantear el tamaño y características del conjunto de datos como un mínimo producto viable (MPV) por lo que se partirá de un volumen reducido de imágenes hasta verificar la viabilidad del experimento a desarrollar. Por tanto, se buscará recopilar entre 100-150 imágenes.

4.3 Adquisición de imágenes

Para construir el dataset se han capturado imágenes de muestras secas y cribadas preparadas con distintas concentraciones del mineral objetivo acompañado del material estéril replicando de esta manera las condiciones reales de exploración minera. Además, se han incluido imágenes de material ausente de monacita junto con otras compuestas únicamente por nódulos de este mineral. Las imágenes que componen el conjunto de datos se han tomado a diferentes escalas, resoluciones e iluminaciones.

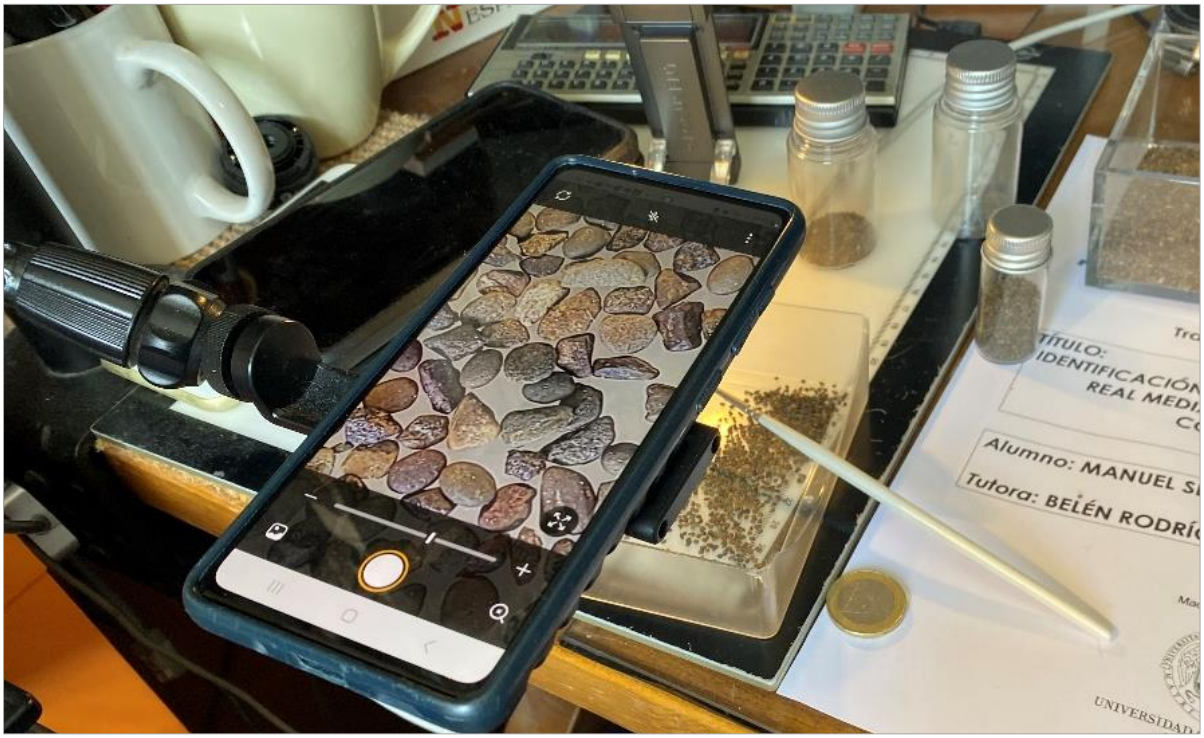


Figura 11: Captura de la imagen de una muestra tomada con el dispositivo móvil S20
Fuente: imagen tomada por el autor

Para fotografiar las muestras se han empleado dos dispositivos, por un lado, un teléfono móvil Samsung Galaxy S20 FE dotado con un sistema de cámaras traseras que incluye un sensor principal de 12 MP con apertura f/1.8, un ultra gran angular de 12 MP y un teleobjetivo de 8 MP con zoom óptico de 3x y zoom digital de hasta 30x. Para manejar mejor la funcionalidad de zoom y enfoque, en este dispositivo se ha instalado la aplicación Magnifying Glass with Flash que utiliza técnicas de aumento digital y combina la cámara del móvil con algoritmos de mejora de imagen para capturar detalles milimétricos

Por otro lado, para conseguir mayor variabilidad de imágenes, también se ha usado una lupa tipo microscopio digital que cuenta con lente dual de 50x y 200x. Con la lente de 50x se ha capturado un subconjunto de imágenes permitiendo así un aumento óptico, lo que aporta robustez al modelo y permite resaltar características microscópicas del mineral como texturas, bordes o patrones cristalinos. Las imágenes capturadas con este sistema son de mayor calidad en términos de resolución y claridad.

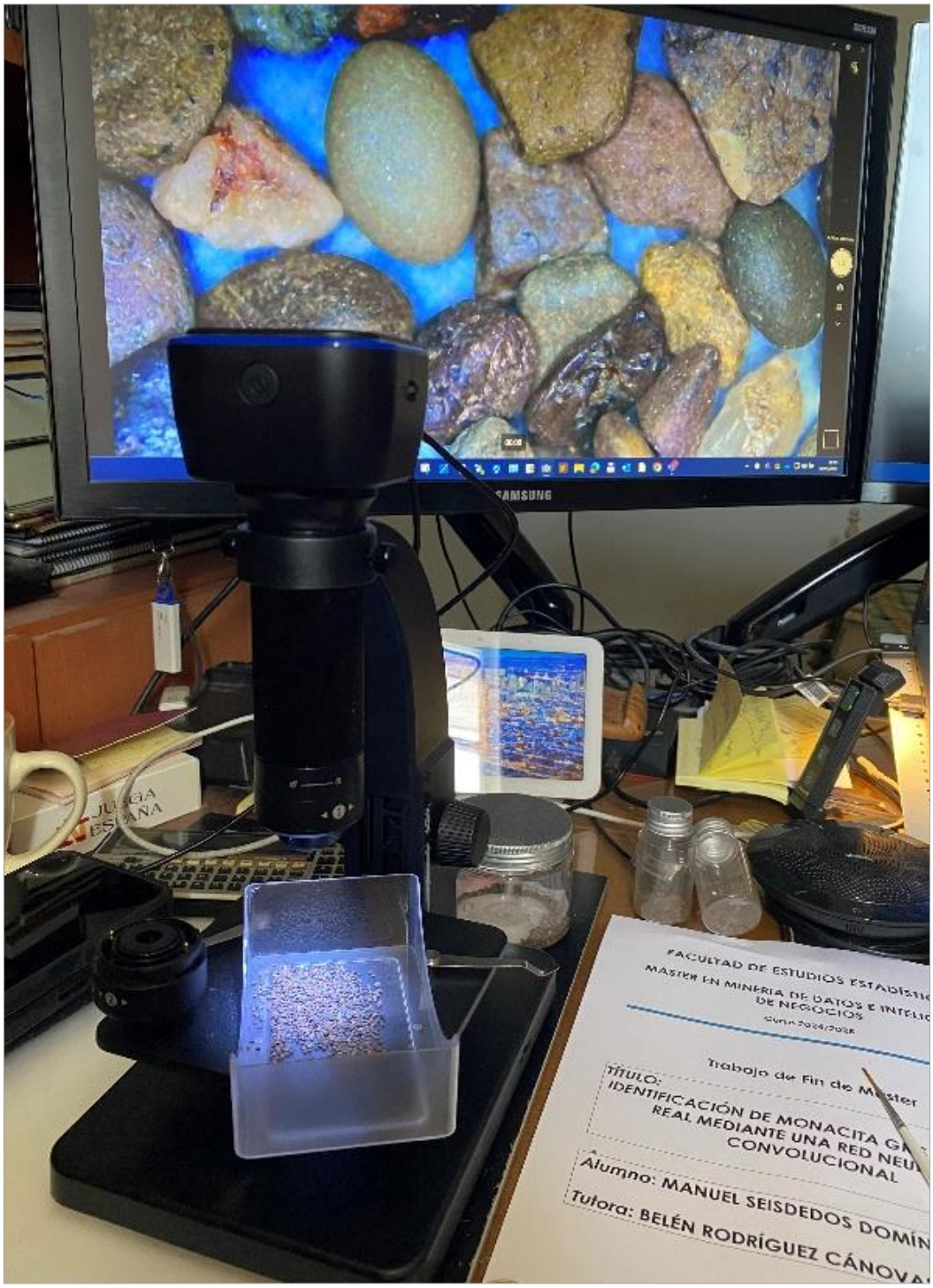


Figura 12: Imagen de una muestra tomada con la lupa digital

Fuente: imagen tomada por el autor

Ambos métodos se complementan para generar un dataset diverso que captura el mineral en diferentes condiciones de escala, iluminación y fondo, lo que mejora el funcionamiento del modelo pensando en que se tendrá que enfrentar a variaciones en el mundo real.

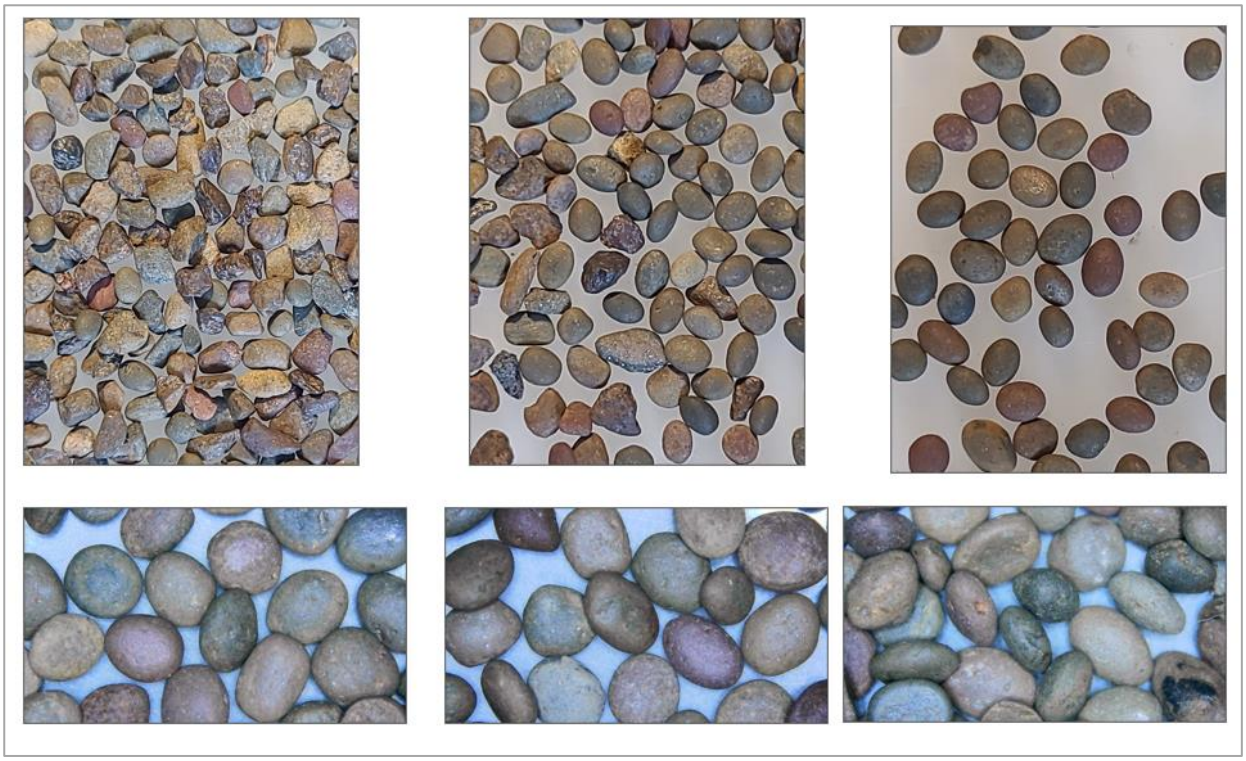


Figura 13: Ejemplos de las imágenes capturadas mediante los dos métodos descritos

4.3 Anotación de imágenes

Únicamente se ha identificado una clase objetivo en las imágenes del dataset, la clase correspondiente a la monacita que se ha nombrado como “Mnz”. Todo lo que no ha sido reconocido como monacita, constituido por otros minerales junto con espacios vacíos, ha quedado sin anotar.

El aplicativo utilizado para crear estas anotaciones ha sido LabelImg, una herramienta de código abierto, escrita en Python, específica para la anotación de imágenes en tareas de visión por ordenador. LabelImg permite crear etiquetas de detección de objetos en formatos compatibles con YOLO.

La instalación de la aplicación se consigue con el comando *pip install labelImg*. Para este TFM, se ha trabajado con la versión 1.8.6.

```
0015256@GM097CPP MINGW64 /c/XYX_REPOS/YOLOv11
$ pip show labelimg
Name: labelImg
Version: 1.8.6
Summary: LabelImg is a graphical image annotation tool and label object bounding boxes in images
Home-page: https://github.com/tzutalin/labelImg
Author: TzuTa Lin
Author-email: tzu.ta.lin@gmail.com
License: MIT license
Location: C:\Users\0015256\AppData\Local\Programs\Python\Python311\Lib\site-packages
Requires: lxml, pyqt5
Required-by:

0015256@GM097CPP MINGW64 /c/XYX_REPOS/YOLOv11
$ labelImg
```

Figura 14: Información del paquete LabelImg versión 1.8.6 mostrada en un terminal bash

Para iniciar el programa se introduce el comando labelImg accediendo a una sencilla interface gráfica desde la que se identifican los objetos mediante rectángulos delimitadores (bounding boxes). Cada cuadro delimitador se asigna a la clase que corresponde, en este caso todas han sido registradas como clase “Mnz”. Las coordenadas de las cajas delimitadoras se almacenan en archivos de texto nombrados idénticamente al fichero de las imágenes que identifican con extensión “.txt”.

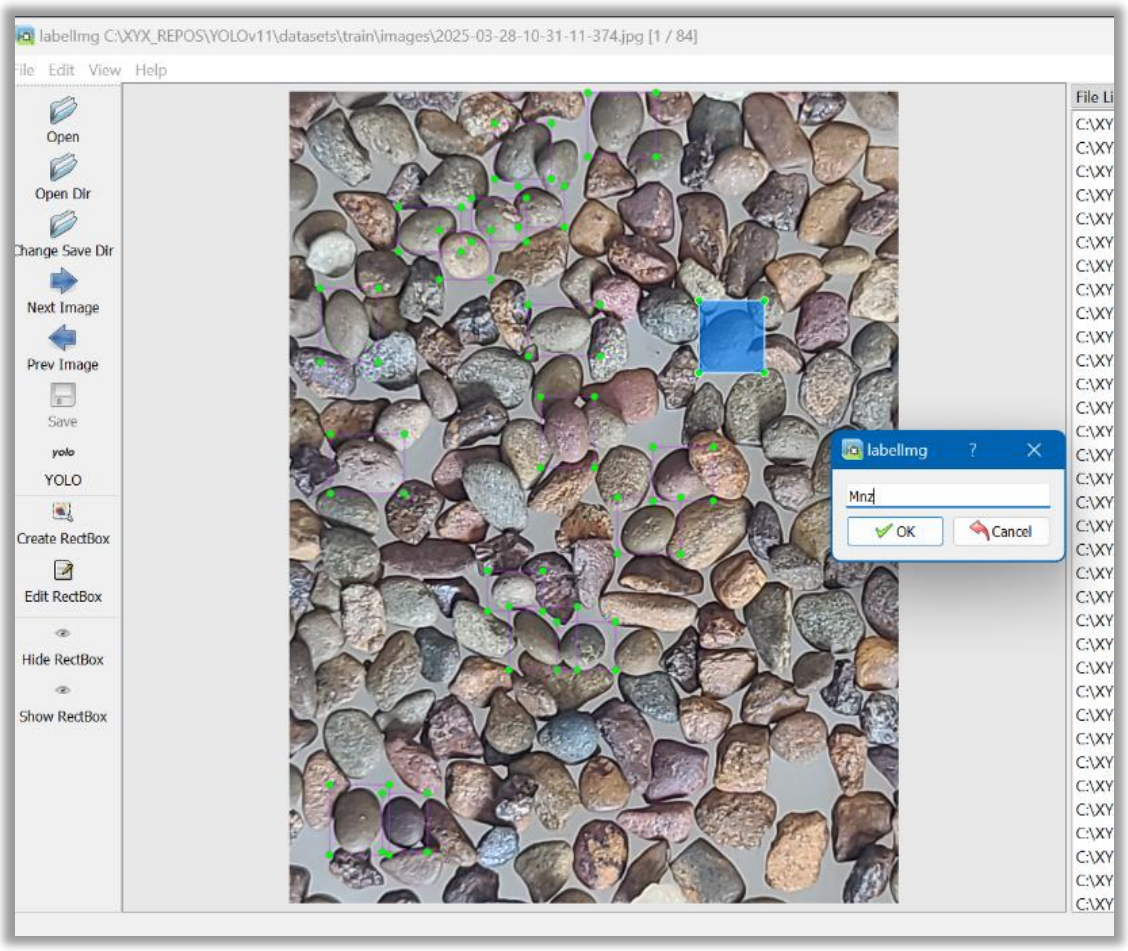


Figura 15: Anotación de una muestra mediante el aplicativo LabelImg

Estos archivos de texto se exportaron a formato YOLO. Estos ficheros especifican las coordenadas normalizadas de cada bounding box junto con la clase correspondiente. Por ejemplo, un archivo de anotación para una imagen tiene el formato:

[“clase” “x_centro” “y_centro” “ancho” “alto”]

Las clases se identifican numéricamente comenzando desde el 0. En la figura 16 se puede ver que la primera columna contiene valores cero, éstos están identificando a la clase 0 que corresponde a la monacita.

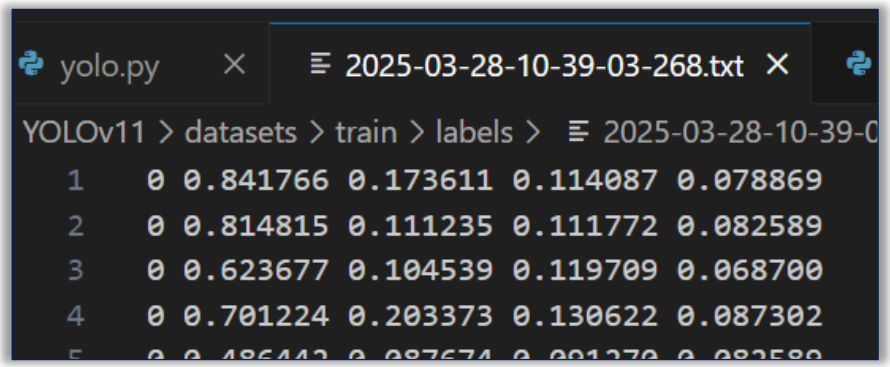


Figura 16: Ejemplo de fichero de coordenadas en formato YOLO

4.4 Partición del dataset

El dataset se ha distribuido en tres carpetas; *train, valid* y *test* donde, a su vez, cada una de ellas se ha dividido en images y labels. Estos datos, describiendo la información relevante del dataset, tiene que registrarse en el fichero de configuración del proyecto, en el caso de este trabajo, este archivo, se ha nombrado como *TFM.yaml* y contiene las rutas a cada partición junto con el número de clases e identificando en una lista el nombre de la clase "Mnz".

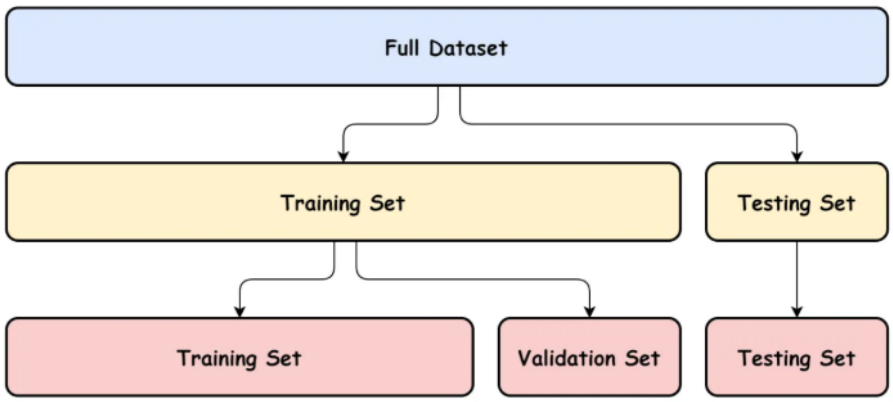


Figura 17: Partición del dataset en CV

Fuente: Ultralytics. (2023)

En la figura 18 se presentan dos capturas que recogen, por un lado, la estructura del dataset y por otro lado el contenido del fichero de configuración descrito.

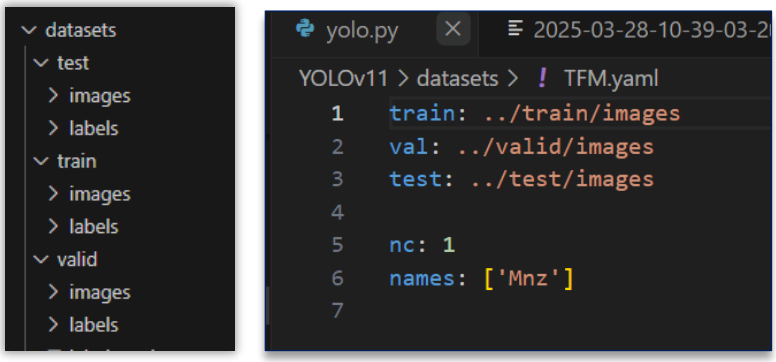


Figura 18: Estructura de directorios del dataset junto con el contenido del archivo TFM.yaml

Referente a la cuantificación de las particiones, hay que distinguir entre número de imágenes anotadas y número de instancias (anotaciones o bounding boxes). La base de datos de imágenes se ha repartido manualmente procurando una relación aproximada de 70% entrenamiento, 20% validación y un 10% test. Esto implica que en todos los entrenamientos, validaciones y testeo han participado las mismas imágenes en sus respectivos roles. Un punto para tratar en futuros trabajos será automatizar el reparto aleatorio de las imágenes y ficheros de anotaciones en sus correspondientes carpetas habiendo parametrizando los porcentajes y priorizando el equilibrio entre imágenes e instancias.

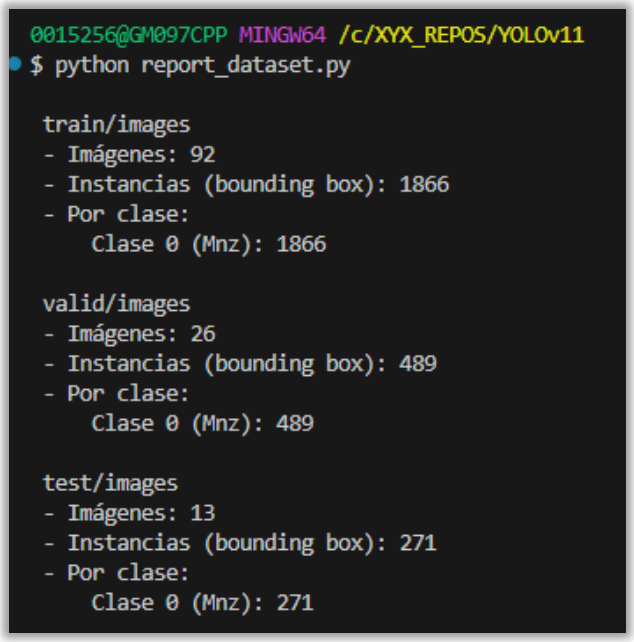


Figura 19: Informe del número de imágenes e instancias del dataset

En total el dataset ha quedado constituido por 131 imágenes que suman 2.626 instancias. La media aritmética de cajas delimitadoras por imagen es de 20.

Como se puede ver en la tabla 3, donde se recogen desglosados los totales de cada partición tanto por el número de imágenes como por el número de instancias, se ha respetado, en ambos casos, la distribución requerida.

	Nº de imágenes		Nº de instancias (bounding boxes)	
train	92	70,23%	1.866	71,06%
valid	26	19,85%	489	18,62%
test	13	9,92%	271	10,32%
Totales	***131***	***100,00%***	***2.626***	***100,00%***

Tabla 3: Distribución del dataset tanto por imágenes como por instancias.

5. ENTRENAMIENTO DEL MODELO

Recordando que el objetivo de este trabajo es conseguir un modelo lo suficientemente fiable y preciso que sea capaz de identificar y cuantificar, en tiempo real, los nódulos de mineral monacita gris; para alcanzarlo se ha diseñado un experimento en dos fases. Estas fases describen cómo se ha entrenado un modelo CNN utilizando una arquitectura YOLO a partir de un modelo base preentrenado y de un conjunto de datos propios ya descrito en el anterior apartado.

En la fase 2 se procederá al entrenamiento y evaluación del modelo final con la configuración obtenida en la fase 1. En la figura 20 se recoge un esquema de las fases del experimento.

Figura 20: Fases del experimento

5.1 Fase 1: exploración de parámetros

En la fase 1 se explorarán los mejores valores para un conjunto de parámetros e hiperparámetros. Esta exploración de parámetros se hará en tres etapas, en la primera etapa se estudiarán los parámetros de YOLO, en una segunda etapa se buscarán los mejores valores para una serie de parámetros base y en la tercera etapa se afinará el modelo mediante entrenamientos con otro grupo de valores para un conjunto de hiperparámetros que resulten interesantes.

5.1.1 Etapa 1: estudio y exploración de los parámetros de entrenamiento

Esta primera etapa corresponde a los trabajos previos a la búsqueda sistemática de los parámetros e hiperparámetros más favorables para optimizar el modelo de detección de objetos ajustado al dataset descrito en el apartado anterior. De una manera informal y no programada se han explorado las variables más relevantes para el entrenamiento.

5.1.1.1 Objetivos de la Etapa 1

Estudiar y seleccionar los parámetros de entrenamiento troncales, junto con otro grupo de hiperparámetros de ajuste, buscando la configuración inicial en la etapa 2 e identificando las variables secundarias con vistas a programar una etapa 3 que afine el modelo.

5.1.1.2 Materiales de la Etapa 1

Hardware:

Ordenador portátil con procesador Intel® Core™ i7-1270P a 2,2GH, la RAM de 32,0 GB y una tarjeta gráfica Intel® Iris® Xe Graphics.

El sistema operativo de 64 bits, Windows 11 Pro versión 23H2

Software:

Usando el entorno Visual Studio Code, con una instalación de Python versión 3.11.2 se ha instalado la librería de ultralytics junto con sus dependencias. La instalación se puede hacer mediante el sistema de instalación de paquetes *pip, pip install ultralytics,* lo que instalará el paquete principal junto con sus dependencias; *matplotlib, numpy, opencv-python, pandas, pillow, psutil, py-cpuinfo, pyyaml, requests, scipy, seaborn, torch, torchvision, tqdm, ultralytics-thop.*

```
0015256@GM097CPP MINGW64 /c/XYX_REPOS/YOLOv11
$ pip show ultralytics
Name: ultralytics
Version: 8.3.107
Summary: Ultralytics YOLO 🚀 for SOTA object detection, multi-object tracking, instance s
egmentation, pose estimation and image classification.
Home-page: https://ultralytics.com
Author:
Author-email: Glenn Jocher <glenn.jocher@ultralytics.com>, Jing Qiu <jing.qiu@ultralytics
.com>
License: AGPL-3.0
Location: C:\Users\0015256\AppData\Local\Programs\Python\Python311\Lib\site-packages
Requires: matplotlib, numpy, opencv-python, pandas, pillow, psutil, py-cpuinfo, pyyaml, r
equests, scipy, seaborn, torch, torchvision, tqdm, ultralytics-thop
```

Figura 21: Características de la instalación de ultralytics y sus dependencias

5.1.1.3 Tareas de la Etapa 1

Se ha consultado bibliografía científica y la documentación de Ultralytics para comprender los principales parámetros de configuración del entorno de entrenamiento

junto con los hiperparámetros, entendiendo como hiperparámetros aquellos que afectan directamente al rendimiento del aprendizaje, disponibles del modelo Yolov11. Igualmente se han estudiado las métricas de validación.

5.1.1.3.1 Parámetros de configuración e hiperparámetros de entrenamiento de Yolov11

Los parámetros, según sus características, se han categorizado en 6 clases: parámetros estructurales, de entrenamiento, de inferencia, aumento de datos, optimización y los referentes a las funciones de pérdida. Esta categorización se puede definir como sigue:

- **Estructurales:** se encuadran las variables básicas. Aquí se clasifican dos parámetros, el tamaño del modelo preentrenado y el fichero de configuración *.yaml.

- **Entrenamiento:** definen datos referentes al tipo de entrenamiento como puede ser el número de épocas o el tamaño de las imágenes con las que trabajar.

- **Inferencia:** referente a los criterios de evaluación como es el índice de confianza.

- **Aumento de datos:** aquí se reúnen los hiperparámetros disponibles para aplicar transformaciones a las imágenes y así enriquecer el conjunto de datos a base de modificaciones del brillo, tono, saturación, construir mosaicos, etc.

- **Optimización:** controlan el optimizador y la tasa de aprendizaje además de técnicas para mejorar la convergencia y evitar el sobreajuste.

- **Perdidas:** ajustan el peso relativo de las diferentes partes de la función de pérdida, como la localización de cajas, la clasificación y la estimación de bordes.

En la tabla 4 se han recogido las variables principales (de configuración e hiperparámetros) disponible para el entrenamiento del modelo.

Categoría de la variable	Parámetros de entrenamiento	Descripción	Valor por defecto	Valores posibles
ESTRUCTURAL	**modelo**	Modelo	Ninguno (debe especificarse)	yolo11[n, s, m, l, x].pt
	data	Ruta al archivo de configuración del dataset (e.g., coco.yaml). Define las clases y rutas de imágenes.	Ninguno (debe especificarse)	Ruta a un archivo YAML válido
ENTRENAMIENTO	**epochs**	Número total de épocas de entrenamiento.	100	[1, 1000] (entero)
	imgsz	Tamaño de la imagen de entrada (ancho y alto, en píxeles).	640	[320, 1280] (múltiplos de 32)
	batch	Tamaño del lote (batch size) por GPU.	16	[1, ∞] (depende de la memoria GPU)
	patience	Épocas a esperar sin mejora en validación antes de detener el entrenamiento.	100	[0, 500] (0 desactiva early stopping)
	project	Directorio donde se guardan los resultados del entrenamiento.	runs/train	Cualquier ruta válida
	verbose	Habilita la salida detallada durante el entrenamiento.	True	[True, False]
	plots	Genera gráficos de métricas y resultados durante el entrenamiento.	True	[True, False]
	device	Dispositivo para entrenar (e.g., CPU, GPU).	None (auto selección)	[cpu, 0, 1, ..., cuda:0, cuda:1]
	amp	Habilita la precisión mixta automática (Automatic Mixed Precision).	True	[True, False]
	workers	Número de subprocesos para cargar datos (DataLoader).	8	[0, 16] (depende de CPU)
	cache	Habilita el almacenamiento en caché de imágenes en RAM/disco.	False	[True, False, disk]
	multi_scale	Habilita el entrenamiento con escalado múltiple de imágenes.	False	[True, False]
	nbs	Tamaño de lote nominal para escalar la pérdida (simula lotes grandes).	64	[16, 128] (múltiplos de 8)
INFERENCIA	**conf**	Umbral de confianza para filtrar detecciones durante la inferencia.	0.25	[0.0, 1.0]

Categoría de la variable	Parámetros de entrenamiento	Descripción	Valor por defecto	Valores posibles
AUMENTO DE DATOS	**mosaic**	Probabilidad de combinar cuatro imágenes en un mosaico.	1.0	[0.0, 1.0]
	mixup	Probabilidad de combinar dos imágenes linealmente.	0.0	[0.0, 1.0]
	scale	Ganancia de escala de la imagen (±).	0.5	[0.0, 1.0]
	hsv_h	Fracción de cambio en el tono.	0.015	[0.0, 0.05]
	hsv_s	Fracción de cambio en la saturación.	0.7	[0.0, 1.0]
	hsv_v	Fracción de cambio en el brillo.	0.4	[0.0, 1.0]
	translate	Fracción de traslación máxima de la imagen.	0.1	[0.0, 0.5]
	degrees	Rotación máxima de la imagen en grados.	0.0	[0.0, 10.0]
	shear	Cizallamiento máximo de la imagen en grados.	0.0	[0.0, 10.0]
	perspective	Fracción de cambio en la perspectiva.	0.0	[0.0, 0.001]
	flipud	Probabilidad de voltear la imagen verticalmente.	0.0	[0.0, 0.5]
	fliplr	Probabilidad de voltear la imagen horizontalmente.	0.5	[0.0, 0.5]
	copy_paste	Probabilidad de copiar y pegar segmentos de una imagen a otra.	0.0	[0.0, 1.0]
	auto_augment	Aplica aumentos automáticos (e.g., RandAugment, ImageNet policies).	None	[None, randaugment, autoaugment]
	erasing	Probabilidad de aplicar borrado aleatorio (Random Erasing).	0.0	[0.0, 0.4]
	crop_fraction	Fracción de la imagen a recortar durante aumentos.	1.0	[0.1, 1.0]
OPTIMIZACIÓN	**lr0**	Tasa de aprendizaje inicial.	0.01 (SGD), 0.001 (AdamW)	[0.0001, 0.1]
	lrf	Fracción de lr0 al final del entrenamiento.	0.01	[0.01, 0.2]

Categoría de la variable	Parámetros de entrenamiento	Descripción	Valor por defecto	Valores posibles
OPTIMIZACIÓN	**cos_lr**	Habilita el programador de tasa de aprendizaje coseno (CosineAnnealingLR).	False	[True, False]
	warmup_epochs	Épocas para aumentar la tasa de aprendizaje desde 0 hasta lr0.	3.0	[0.0, 5.0]
	warmup_bias_lr	Tasa de aprendizaje inicial para sesgos durante el calentamiento.	0.1	[0.0, 0.2]
	weight_decay	Penalización que evita que los pesos se vuelvan demasiado grandes (regularización)	0.0005	[0.0, 0.001]
	optimizer	Optimizador utilizado (e.g., SGD, Adam, AdamW).	SGD	[SGD, Adam, AdamW]
	momentum	Media móvil exponencial de los gradientes (para SGD, o beta1 para AdamW).	0.937 (SGD), 0.9 (AdamW)	[0.8, 0.999]
PÉRDIDAS	**box**	Ponderación de la pérdida de cajas delimitadoras.	0.05	[0.02, 0.2]
	cls	Ponderación de la pérdida de clasificación.	0.5	[0.3, 0.9]
	dfl	Ponderación de la pérdida de distribución focal (Distribution Focal Loss) para cajas.	1.5	[0.0, 2.0]
	iou	Umbral de IoU para asignar anclajes	0.20	[0.1, 0.7]

Tabla 4: Configuración de los parámetros e hiperparámetros más relevantes en YOLOv11

5.1.1.3.2 Modelos preentrenados de YOLO

Los modelos YOLO disponibles en formato *.pt han sido preentrenados inicialmente sobre el conjunto de datos COCO (Common Objects in Context), (tabla 1). Este dataset de amplia difusión contiene 330.000 imágenes de las que 200.000 tienen anotaciones para tareas de detección. Son 80 las categorías de objetos incluyendo objetos muy comunes como coches, bicicletas y animales, y otras más específicas como paraguas, bolsos y material deportivo.

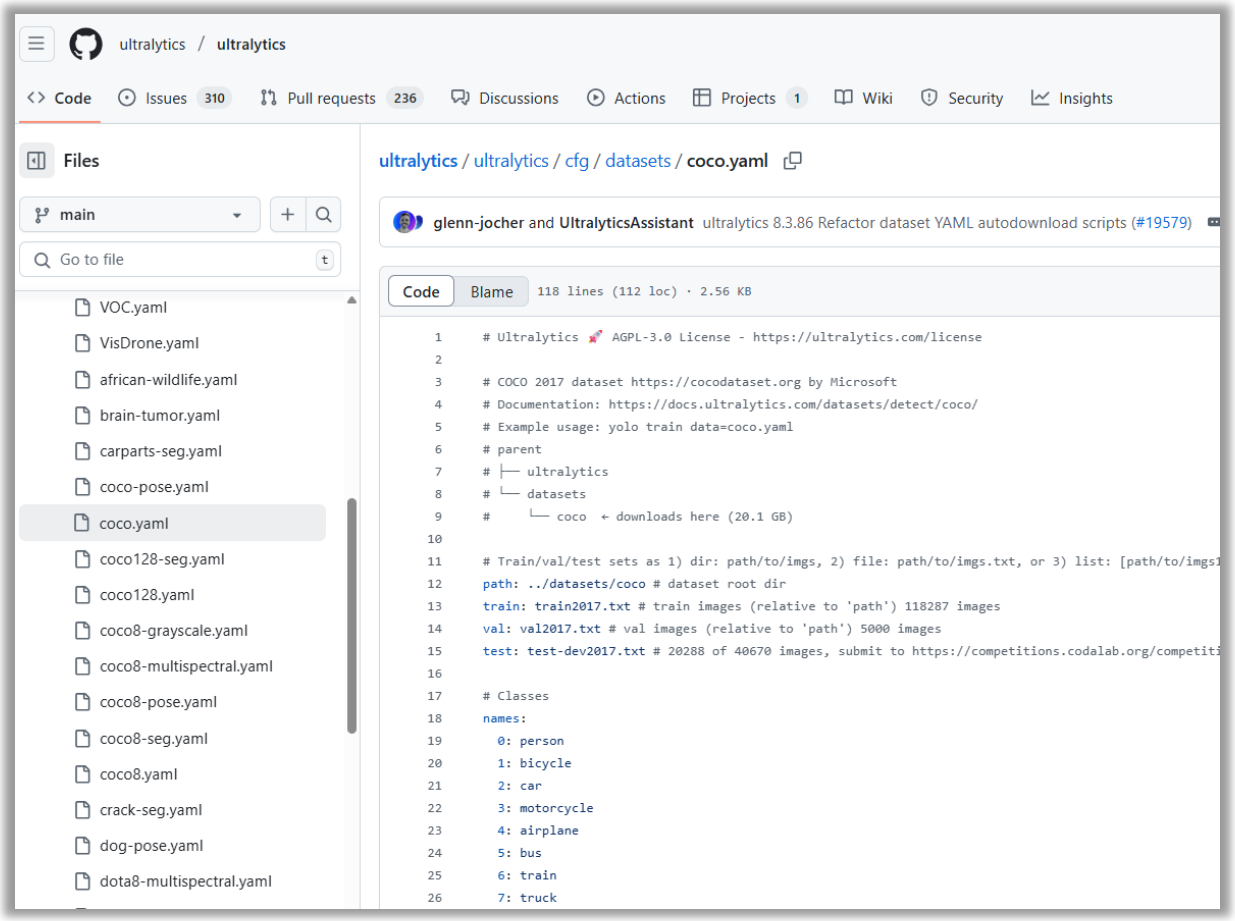


Figura 22: Captura de la imagen del fichero coco.yaml en Github

Fuente: *https://github.com/ultralytics/ultralytics/blob/main/ultralytics/cfg/datasets/coco.yaml*

Como resultado de este entrenamiento hay disponibles cinco versiones del modelo YOLO según su complejidad, n (nano), s (small), m (médium), l (large) y x (extra-large), diferenciándose, fundamentalmente, por el número de parámetros de entrenamiento, desde los 2,6 millones del nano hasta los casi 60 millones del mayor de ellos. En la figura 23 se recogen las principales características de cada modelo.

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed T4 TensorRT10 (ms)	params (M)	FLOPs (B)
YOLO11n	640	39.5	56.1 ± 0.8	1.5 ± 0.0	2.6	6.5
YOLO11s	640	47.0	90.0 ± 1.2	2.5 ± 0.0	9.4	21.5
YOLO11m	640	51.5	183.2 ± 2.0	4.7 ± 0.1	20.1	68.0
YOLO11l	640	53.4	238.6 ± 1.4	6.2 ± 0.1	25.3	86.9
YOLO11x	640	54.7	462.8 ± 6.7	11.3 ± 0.2	56.9	194.9

Figura 23: Características de las diferentes versiones de los modelos YOLO

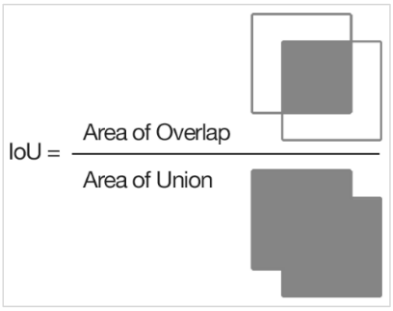
Fuente: *https://docs.ultralytics.com/datasets/detect/coco/*

Como se aprecia en la figura 23, el tamaño del modelo elegido influirá en el tamaño del modelo entrenado y a medida que se incrementa el número de variables crecen igualmente las necesidades computacionales requeridas para ejecutarlos. De esta manera, modelos más ligeros (n y s) están diseñados para entornos con recursos limitados, como dispositivos móviles o sistemas embebidos, ya que requieren menos memoria, menor potencia de cálculo y ofrecen inferencias más rápidas, aunque a costa de una ligera pérdida de precisión. Por el contrario, los modelos más grandes (m, l, x) alcanzan mayores niveles de exactitud, pero demandan recursos computacionales superiores, tanto en GPU como en CPU, lo que limita su ejecución a entornos con hardware especializado, por ejemplo, estaciones de trabajo con GPU dedicadas o servidores en la nube.

5.1.1.3.3 Métricas de YOLO

La evaluación de modelos para la detención de objetos requiere un enfoque distinto al ser utilizado, normalmente, en la clasificación de imágenes. Mientras que en esta última es suficiente verificar si la categoría predicha coincide, para la detención es igual de importante saber la ubicación exacta del objeto dentro de la imagen. Además, se necesita saber cómo de precisa es esa ubicación. Por lo tanto, se emplean métricas específicas que consideran tanto la clasificación como la precisión espacial de las predicciones.

Una de las medidas esenciales en esta área es el IoU (Intersection over Union), que evalúa el grado de coincidencia entre la región delimitadora pronosticada por el modelo y la región real (verdad real). Basándose en este valor inicialmente establecido se considera el mAP (mean Average Precision), que muestra la precisión promedio del modelo en diversas categorías y niveles específicos del Índice de Intersección sobre Unión (IoU).



mAP50-95: mean Average Precision promediada sobre umbrales de IoU desde 0,5 hasta 0,95. Evalúa la precisión del modelo en diferentes niveles de exigencia para la localización de objetos (más estricto cuanto mayor es el IoU).

Es la métrica principal para comparar modelos ya que ofrece una visión global del rendimiento del modelo en distintos niveles de dificultad de detección.

- **mAP50**: precisión media calculada con un umbral de IoU de 0,5. Considera una detección correcta si el solapamiento entre el cuadro predicho y el real es al menos 50%. Es menos estricto, enfocado en detecciones "fáciles".

- **Verdaderos Positivos (TP):** el modelo predice monacita correctamente.

- **Verdaderos Negativos (TN):** el modelo predice correctamente que la imagen no se corresponde a la clase monacita

- **Falsos Positivos (FP):** el modelo predice erróneamente que una imagen pertenece a una clase, en el caso del TFM, el modelo devolverá un mineral distinto a la monacita como si lo fuese.

- **Falsos Negativos (FN):** el modelo predice incorrectamente que un individuo no pertenece a una clase, en este caso habrá nódulos de monacita que se pasan por alto. A efectos prácticos para el caso de uso de este TFM es la métrica más desfavorable.

- **Precision**: proporción de detecciones correctas (verdaderos positivos) respecto al total de detecciones realizadas (verdaderos positivos + falsos positivos). Indica cuántas detecciones fueron correctas.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall**: se refiere a la proporción de objetos reales detectados correctamente (verdaderos positivos) respecto al total de objetos presentes (verdaderos positivos + falsos negativos). Mide la capacidad del modelo de encontrar todas las instancias de los objetos en las imágenes.

$$Recall = \frac{TP}{TP + FN}$$

- **F1 Score**: es una métrica que combina precision y recall para evaluar el equilibrio entre ambas en el modelo de detección. Se calcula como la media armónica:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

El rango se encuentra entre 0 a 1, donde 1 indica un balance perfecto entre precisión (detecciones correctas respecto al total de predicciones) y recall (objetos detectados respecto al total de objetos reales). Es útil cuando se busca un equilibrio entre minimizar falsos positivos (alta precisión) y falsos negativos (alto recall).

Se ha elaborado una tabla (tabla 5) de clasificación cualitativa del modelo a partir de los rangos de valores de las métricas que se obtengan al evaluar los modelos.

	mAP50-95	mAP50	Precision	Recall	F1 Score	Descripción
Excelente	> 0,90	> 0,95	> 0,95	> 0,95	> 0,95	**Optimización extrema, comparable al estado del arte**
Muy bueno	0,75-0,9	0,9-0,95	0,9-0,95	0,9-0,95	0,9-0,95	**Aplicaciones prácticas, alta generalización**
Bueno	0,6-0,75	0,7-0,9	0,8-0,9	0,8-0,9	0,8-0,9	**Rendimiento sólido, propio de datasets pequeños**
Aceptable	0,4-0,6	0,5-0,7	0,6-0,8	0,6-0,8	0,6-0,8	**Tareas básicas. Mejorable**
Bajo	< 0,4	< 0,5	< 0,5	< 0,5	< 0,5	**Pobre desempeño, requiere ajustes**

Tabla 5: Clasificación cualitativa del desempeño del modelo en función de métricas resultantes
Fuente: *elaboración propia*

5.1.1.3.4 Entrenamientos preliminares

Una vez determinados estos parámetros se procederá a realizar entrenamientos preliminares desde local. El objeto es testear el comportamiento del hardware respecto a los valores de los parámetros a estudiar.

5.1.1.4 Conclusiones de la Etapa 1

1.- Elección de variables a explorar: después del análisis realizado en esta primera etapa se han seleccionado tres grupos de variables:

A **Variables fijas**: se trata de un primer subconjunto que, por su importancia y funcionamiento, según el caso de uso planteado, se les asignará unos valores fijos durante el experimento, valores que no coinciden con los asignados por defecto en el aplicativo. Estas variables son:

- **optimizer= "AdamW":** según la documentación de ultralitics, esta elección es común en la arquitectura YOLOv11 ya que balancea velocidad y precisión en tareas de visión por ordenador además de contribuir a evitar el sobreajuste.

- **weight_decay= 0.001:** decaimiento de pesos, manteniendo valores medios en este parámetro se penaliza los pesos grandes en la red reduciendo, por tanto, el riesgo de sobreajuste del modelo.

- **cos_lrc= True:** activando cosine learning rate decay, decaimiento de la tasa de aprendizaje del coseno, permite una exploración inicial amplia del espacio de características, seguida de un refinamiento preciso, optimizando la localización y clasificación de objetos pequeños en imágenes complejas.

B **Variables base a explorar en la etapa 2**: por otro lado, los parámetros base seleccionados son:

- **modelo = ['s', 'm', 'l']:** de los cinco modelos preentrenados disponibles se descartan los dos extremos, el menor y el mayor. Se probarán los tres que tienen una complejidad media en cuanto a computación y arquitectura. En el punto 5.1.1.3.2 se han descrito los distintos modelos e indicado que a mayor tamaño del modelo mayores requerimientos de computación buscando mejores precisiones. Esto es importante puesto que se enfoca el experimento a conseguir las mejores métricas a consta de rebajar las opciones funcionales a la hora de ejecutar el modelo resultante.

- **imgz = [896, 1280]:** al tratarse de minerales milimétricos se descarta la menor resolución de 640 píxeles, configurada por defecto, y se testearán imágenes de 896 y 1280 píxeles respectivamente.

- **bach = [8, 16, 32]:** para valorar el efecto del tamaño de lote en la estabilidad y velocidad de convergencia del entrenamiento se probarán estos tres valores.

- **lr0 = [0.001, 0.0005]:** se comprobarán dos valores para analizar el comportamiento de la tasa de aprendizaje inicial.

Figura 24: Esquema de la selección de variables y valores de exploración para las etapas 2 y 3 de la fase 1.

C **Hiperparámetros a explorar en la Etapa 3**: se han elegido siete hiperparámetros para analizar dos valores de cada uno de ellos. Consultando la tabla 4 se puede ver que seis de ellos pertenecen a la categoría correspondiente al aumento de datos, mientras que box, pertenece a los hiperparámetros que controlan las pérdidas en la precisión de las cajas delimitadoras. Los valores a probar son:

- **mosaic = [0.5, 1.0]**
- **mixup = [0.0, 0.5]**
- **scale = [0.2, 0.5]**
- **hsv_h = [0.015, 0.05]**
- **hsv_s = [0.4, 0.7]**
- **hsv_v = [0.4, 0.7]**
- **box = [0.05, 0.2]**

2.- Con el hardware descrito, los únicos modelos preentrenados que se pueden probar, dentro de unos límites de tiempo de entrenamiento razonable, son YOLO11n.pt y YOLO11s.pt. Se trata de un parámetro de configuración estructural que resulta clave para mejorar los resultados de precisión y exactitud perseguidos, por lo que se descarta trabajar desde local y se opta por continuar las siguientes fases en un entorno que disponga de mejores recursos de computación.

5.1.2 Etapa 2: entrenamiento base

En esta fase se ejecutarán los entrenamientos definidos en la fase anterior y se analizarán los valores óptimos de las variables seleccionadas para que participen en la siguiente etapa.

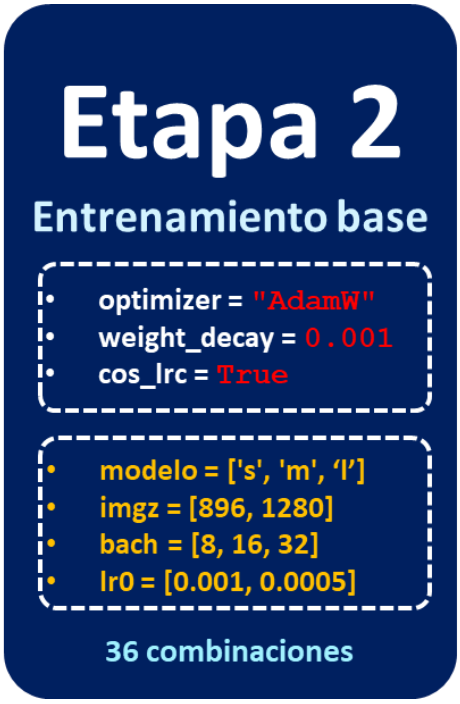


Figura 25: Esquema del entrenamiento de la Etapa 2

5.1.2.1 Objetivos de la Etapa 2

El objetivo consiste en identificar los valores óptimos de las variables que se exploran en esta etapa.

5.1.2.2 Materiales de la Etapa 2

Para el entrenamiento del modelo se ha optado por hacerlo desde Google Colab Pro en un entorno Python-3.11.13 torch-2.6.0+cu124 CUDA:0 con acceso a NVIDIA A100-SXM4-40GB, 40507MiB)

Se ha utilizado un disco montado en Google Drive para persistir los modelos, logs y dataframes.

Las librerías de Python empleadas son además de ***ultralytics*** *y sus dependencias:*

- ***pandas:*** *para manejar los dataframe de almacenamiento de resultados.*
- ***itertools:*** *usado para la gestión de las combinaciones de los parámetros.*
- ***google.colab****: para interaccionar con Google Drive.*
- ***shutil y os****: se han usado para gestionar archivos y directorios.*

Para generar las salidas gráficas en los scripts que analizan las métricas adicionalmente se han instalado.

- ***matplot*** *y* ***seaborn:*** *librerías para visualizar datos.*

5.1.2.3 Tareas de la Etapa 2

5.1.2.3.1 Automatizar el entrenamiento de las 36 combinaciones definidas

Se automatiza mediante un script de Python que ejecuta los entrenamientos de esta fase. En esta etapa se entrenarán 50 épocas para cada entrenamiento. El código fuente se incluye como **Anejo I**.

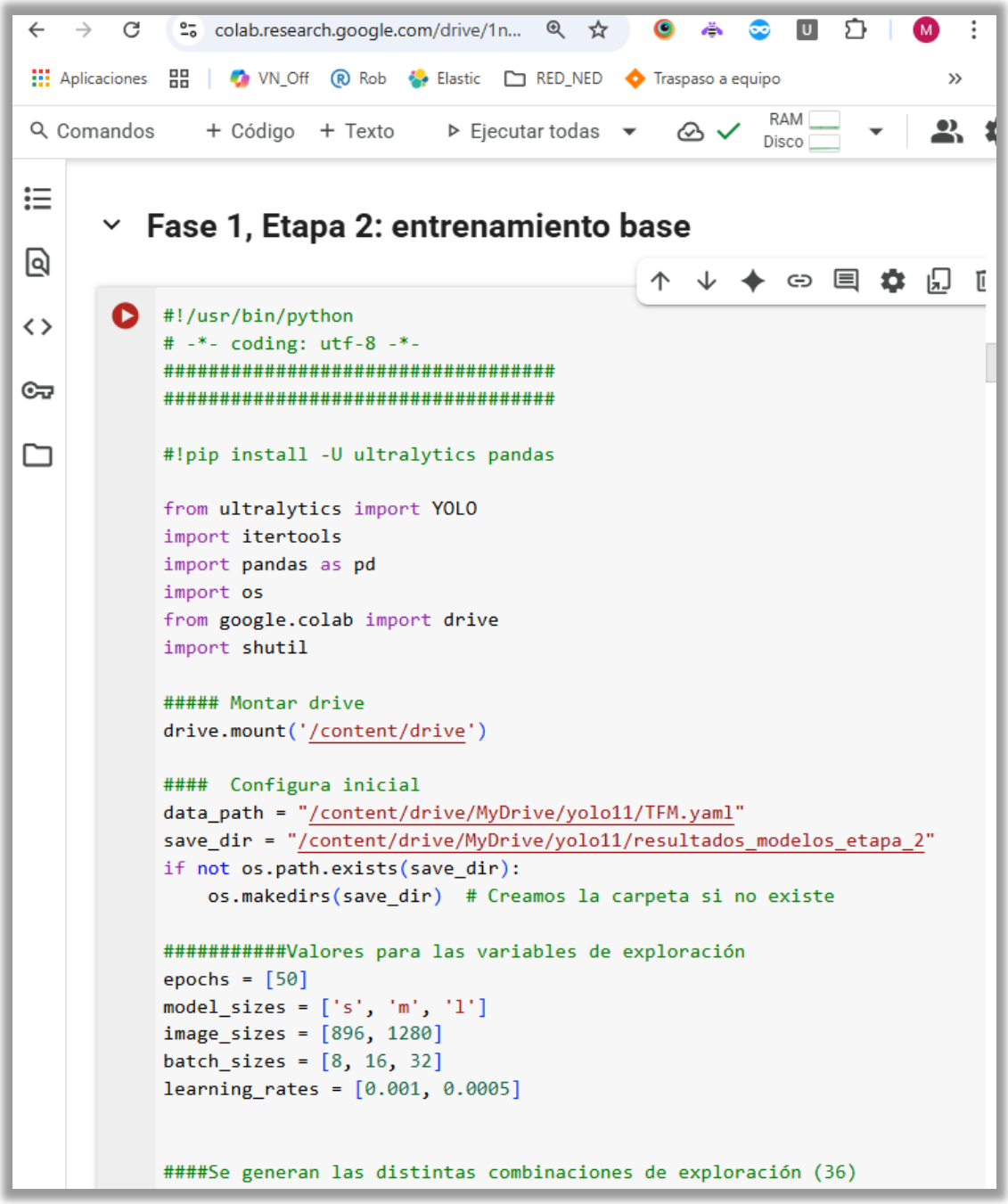


Figura 26: Captura del script de entrenamiento de la etapa 2

Los resultados de las métricas se almacenan en el fichero *resultados_etapa_2.csv*

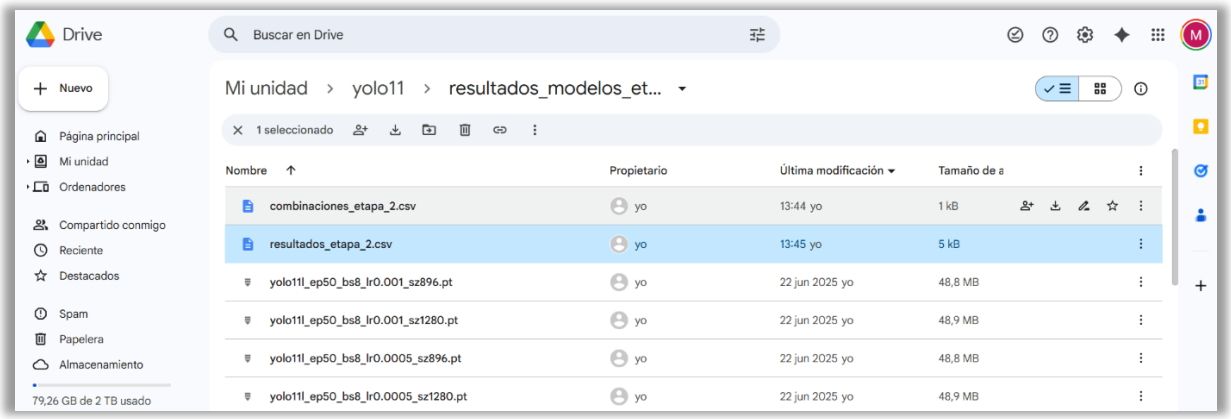
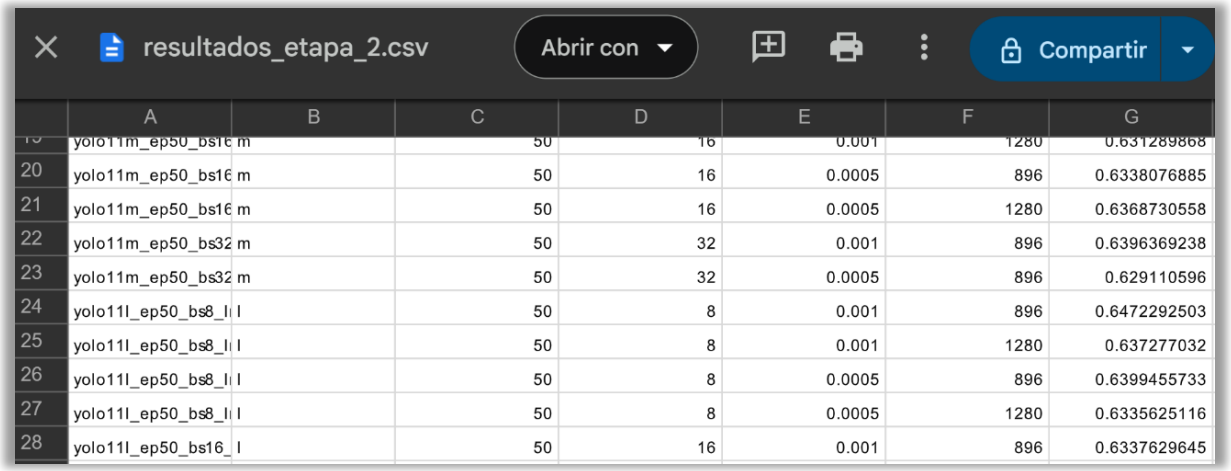


Figura 27: Vista de la carpeta de resultados de la Etapa 2 en Google Drive



	A	B	C	D	E	F	G
19	yolo11m_ep50_bs16	m	50	16	0.001	1280	0.631289868
20	yolo11m_ep50_bs16	m	50	16	0.0005	896	0.6338076885
21	yolo11m_ep50_bs16	m	50	16	0.0005	1280	0.6368730558
22	yolo11m_ep50_bs32	m	50	32	0.001	896	0.6396369238
23	yolo11m_ep50_bs32	m	50	32	0.0005	896	0.629110596
24	yolo11l_ep50_bs8_l	l	50	8	0.001	896	0.6472292503
25	yolo11l_ep50_bs8_l	l	50	8	0.001	1280	0.637277032
26	yolo11l_ep50_bs8_l	l	50	8	0.0005	896	0.6399455733
27	yolo11l_ep50_bs8_l	l	50	8	0.0005	1280	0.6335625116
28	yolo11l_ep50_bs16_	l	50	16	0.001	896	0.6337629645

Figura 28: Vista del fichero de salida que recoge los resultados de la etapa 2.

5.1.2.3.2 Analizar y discutir los valores óptimos

De las 36 ejecuciones han fallado las cuatro combinaciones que incluían un batch de 32 junto con resolución de 1280 píxeles, estos fallos se deben a que la tarjeta GPU A100 no ha conseguido concluir estos entrenamientos por lo que se han obviado en el análisis centrando el estudio en los 32 modelos restantes.

Con las métricas de esos 32 modelos se han generado dos gráficos Scatterplot donde se representan los resultados de los modelos preentrenados en relación a las métricas recall y precisión frente a mAP50-95, esto es, métricas que identifican los falsos negativos y falsos positivos respectivamente. Como se puede ver en la figura 29, el modelo Yolo11l.pt es el que mejores resultados devuelve para maximizar mAP50-95.

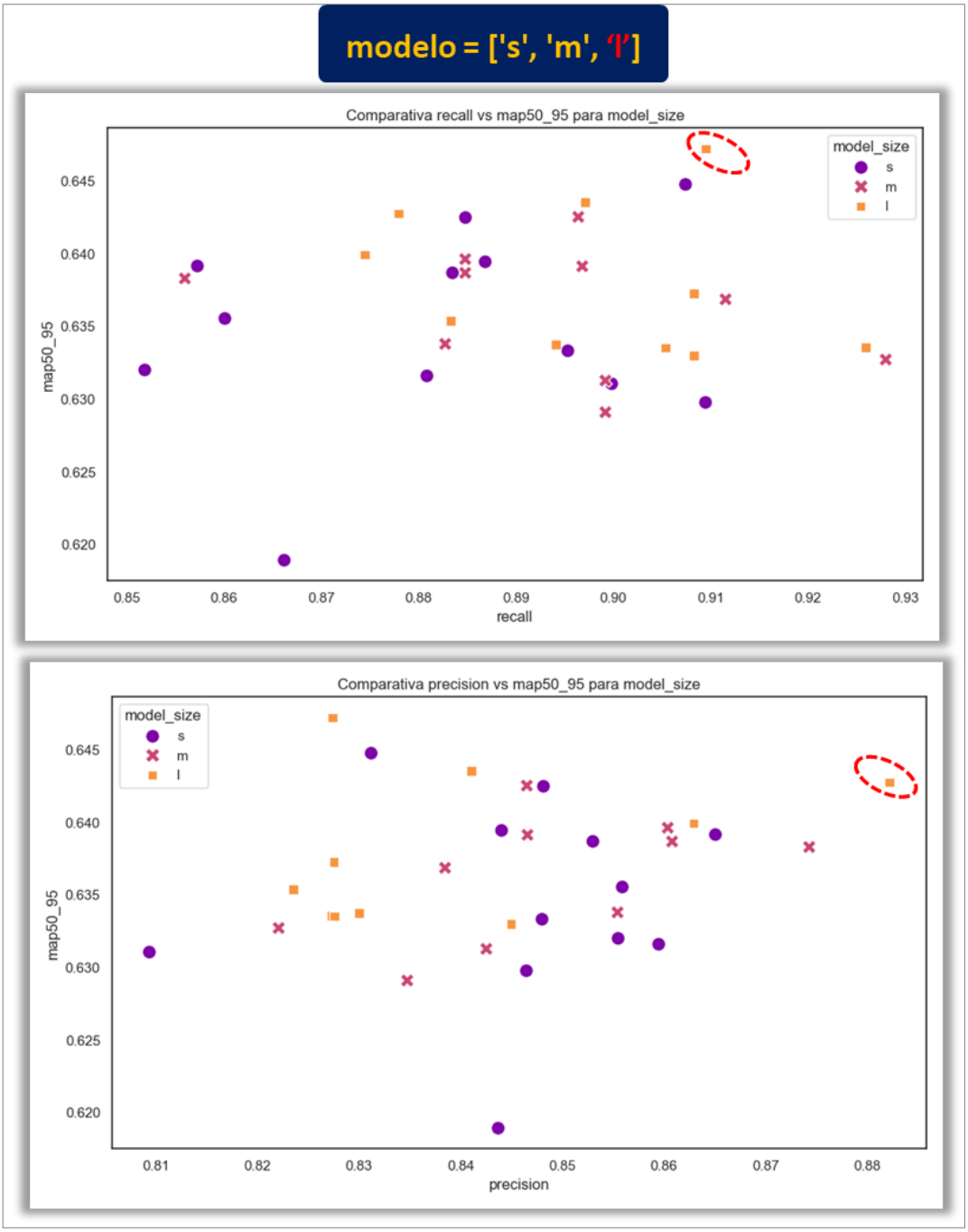


Figura 29: Scatterplot que enfrenta las métricas precisión y recall con mAP50-95 y el tamaño del modelo

Igualmente se han representado en gráficos tipo boxplot (figura 30) las cuatro variables analizadas respecto de la métrica mAP50-95. En el caso del parámetro modelo, se reafirma la elección ya hecha de "l"; se aprecia la mediana por encima de los otros modelos. De la misma manera se ha seleccionado 896 píxeles, 8 batch y 0,001 para la tasa de aprendizaje.

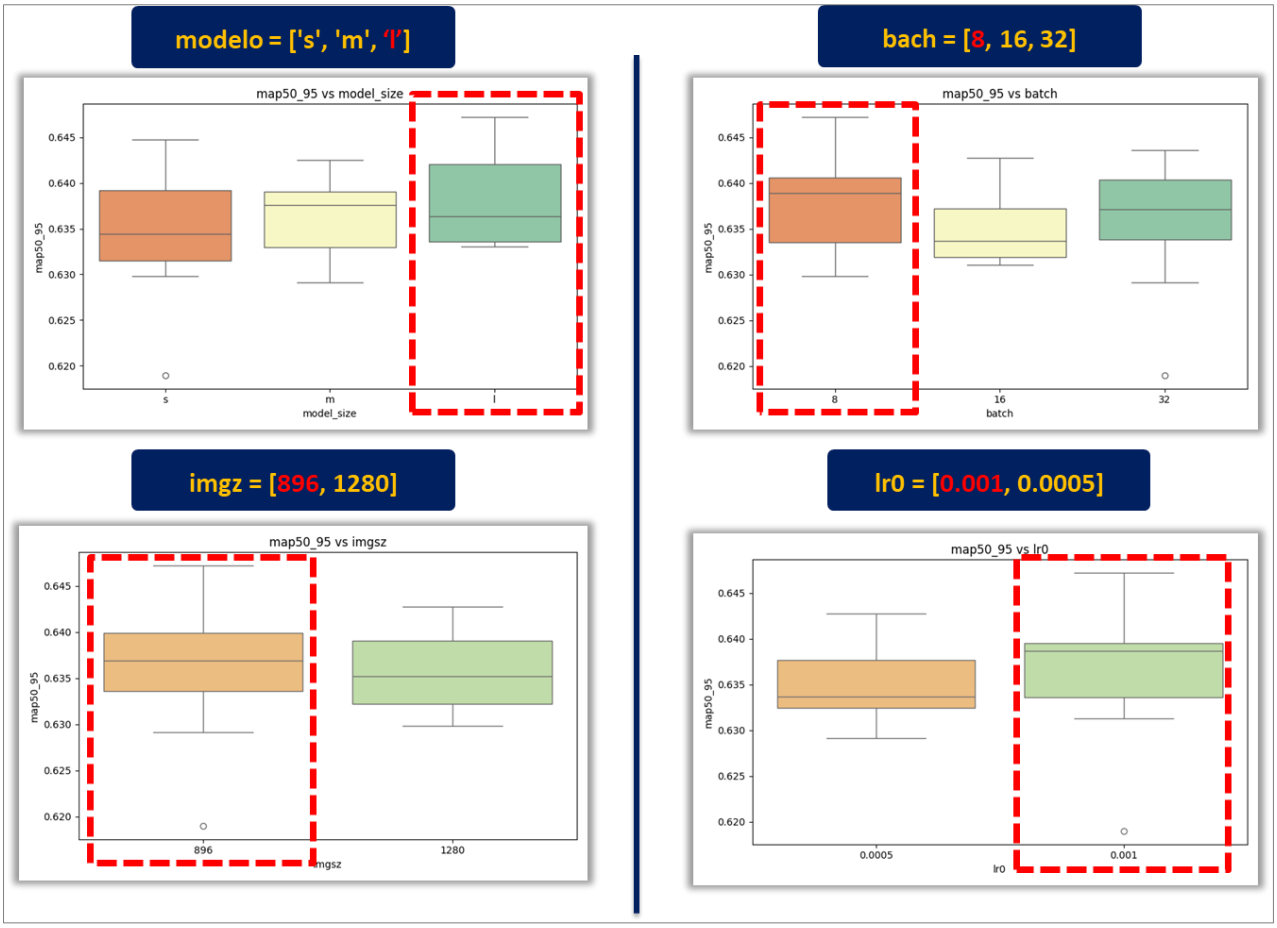


Figura 30: Gráficos boxplot de las métricas de mAP50-95 para el modelo, la resolución de imágenes, bach y lr0

5.1.2.4 Conclusiones de la Etapa 2

Como resultado del entrenamiento base se concluyen que los valores de los cuatro parámetros estudiados en esta etapa son los que se muestran en color rojo en la figura 31.

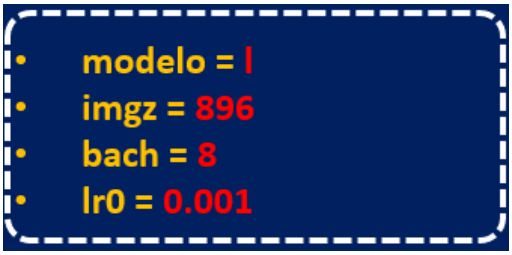


Figura 31: Los valores elegidos en la etapa 2 marcados en color rojo

Por otro lado, las métricas que corresponden al mejor modelo de esta etapa, *yolo11l_ep50_bs8_lr0.001_sz896.pt*, para un índice de confianza del 0,40, se recogen en la tabla 6.

- 50 épocas - conf=0,40 - Validación	Etapa 2 (v)
mAP50-95	0,6412
mAP50	0,8921
Precision	0,8537
Recall	0,8644
F1 Score	0,8590

Tabla 6: Métricas del modelo seleccionado en esta etapa.

5.1.3 Etapa 3: afinamiento del modelo

En esta última etapa de la fase 1 se ejecutará la exploración de los hiperparámetros definidos en la etapa 1. En la figura 32 se recogen estas variables, junto con los valores que se testearán en color amarillo.

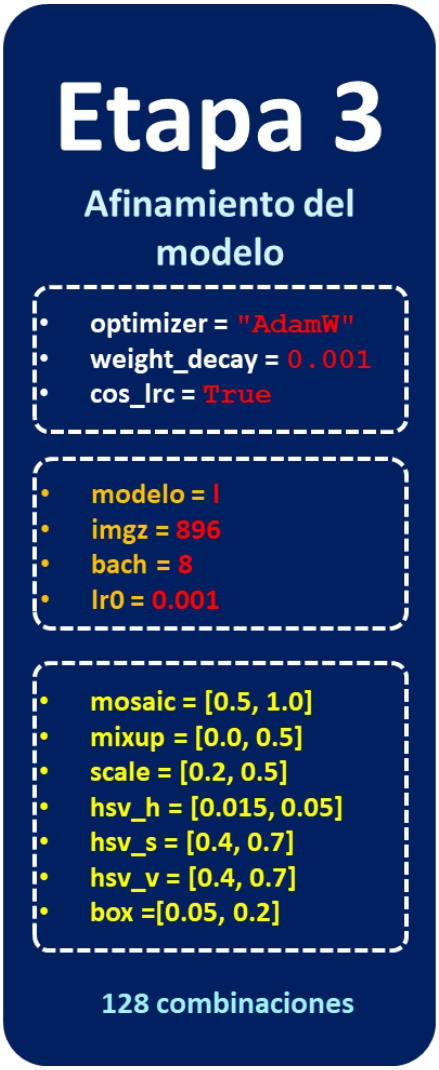


Figura 32: Esquema con los hiperparámetros estudiados en la etapa 3

5.1.3.1 Objetivos de la Etapa 3

Mejorar las métricas obtenidas en la etapa 2 incrementando la exactitud y precisión del modelo.

5.1.3.2 Materiales de la Etapa 3

Los materiales de esta etapa son los mismos que en la anterior; para el entrenamiento del modelo se utilizó Google Colab Pro en un entorno Python-3.11.13 torch-2.6.0+cu124 CUDA:0 con acceso a NVIDIA A100-SXM4-40GB, 40507MiB)

Se ha utilizado un disco montado en Google Drive para persistir los modelos, logs y dataframes.

5.1.3.3 Tareas de la Etapa 3

5.1.3.3.1 Automatizar los entrenamientos

Se automatiza mediante un script de Python los entrenamientos propuestos en esta etapa. Por un lado, se resolverán las posibles combinaciones estudiadas para, a continuación, ejecutar los entrenamientos. Son siete variables con dos valores por variable, por lo que las combinaciones posibles son 128. Se ha incluido en el script la posibilidad de definir el número de entrenamientos pudiendo hacerlo sobre una muestra aleatoria. Esta fase exploratoria se realiza sobre la mitad de las combinaciones posibles, esto es, se completan 64 entrenamientos, y para 100 épocas por entrenamiento. El código fuente se incluye como **Anejo II**.

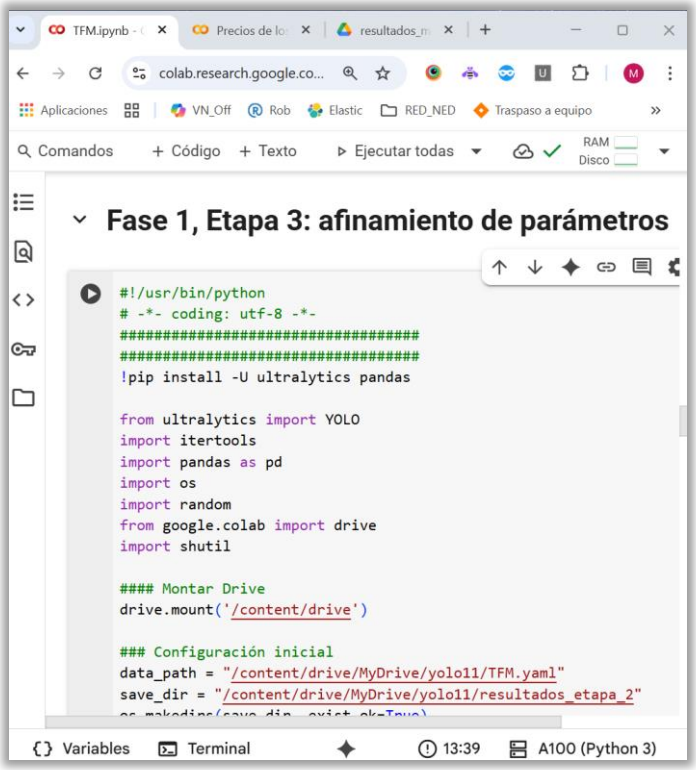
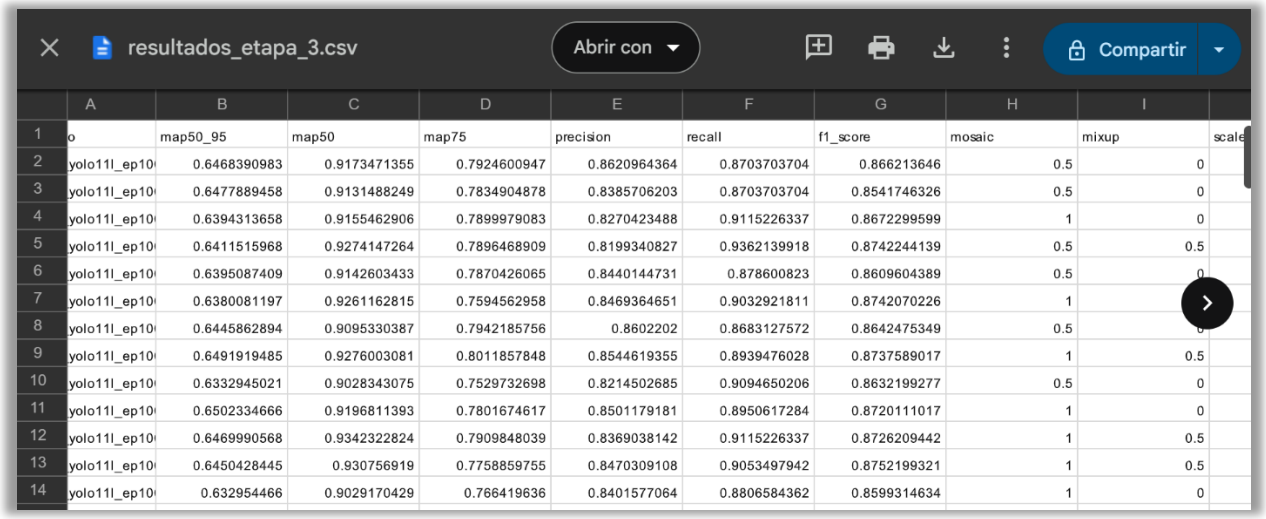


Figura 33: Captura del script de entrenamiento de la etapa 3

5.1.3.3.2 Analizar y discutir los valores de la Etapa 3

Según el criterio de seleccionar el mejor modelo según la métrica mAP50-95 obtenemos que el mejor entrenamiento ha sido:
yolo11l_ep100_bs8_lr0.001_sz896_mosaic0.5_mixup0.5_scale0.5_hsv_h0.015_hsv_s0.7_hsv_v0.4_box0.2.pt



resultados_etapa_3.csv — Abrir con — Compartir

	A	B	C	D	E	F	G	H	I	
1	o	map50_95	map50	map75	precision	recall	f1_score	mosaic	mixup	scale
2	yolo11l_ep10	0.6468390983	0.9173471355	0.7924600947	0.8620964364	0.8703703704	0.866213646	0.5	0	
3	yolo11l_ep10	0.6477889458	0.9131488249	0.7834904878	0.8385706203	0.8703703704	0.8541746326	0.5	0	
4	yolo11l_ep10	0.6394313658	0.9155462906	0.7899979083	0.8270423488	0.9115226337	0.8672299599	1	0	
5	yolo11l_ep10	0.6411515968	0.9274147264	0.7896468909	0.8199340827	0.9362139918	0.8742244139	0.5	0.5	
6	yolo11l_ep10	0.6395087409	0.9142603433	0.7870426065	0.8440144731	0.878600823	0.8609604389	0.5	0	
7	yolo11l_ep10	0.6380081197	0.9261162815	0.7594562958	0.8469364651	0.9032921811	0.8742070226	1	0	
8	yolo11l_ep10	0.6445862894	0.9095330387	0.7942185756	0.8602202	0.8683127572	0.8642475349	0.5	0	
9	yolo11l_ep10	0.6491919485	0.9276003081	0.8011857848	0.8544619355	0.8939476028	0.8737589017	1	0.5	
10	yolo11l_ep10	0.6332945021	0.9028343075	0.7529732698	0.8214502685	0.9094650206	0.8632199277	0.5	0	
11	yolo11l_ep10	0.6502334666	0.9196811393	0.7801674617	0.8501179181	0.8950617284	0.8720111017	1	0	
12	yolo11l_ep10	0.6469990568	0.9342322824	0.7909848039	0.8369038142	0.9115226337	0.8726209442	1	0.5	
13	yolo11l_ep10	0.6450428445	0.930756919	0.7758859755	0.8470309108	0.9053497942	0.8752199321	1	0.5	
14	yolo11l_ep10	0.632954466	0.9029170429	0.766419636	0.8401577064	0.8806584362	0.8599314634	1	0	

Figura 34: Vista del fichero de resultados de esta etapa 3

5.1.3.4 Conclusiones de la Etapa 3

Como resultado de esta fase de afinamiento del modelo, se concluye que los valores de los siete hiperparámetros estudiados en esta etapa son los que se muestran en color rojo en la figura 35.

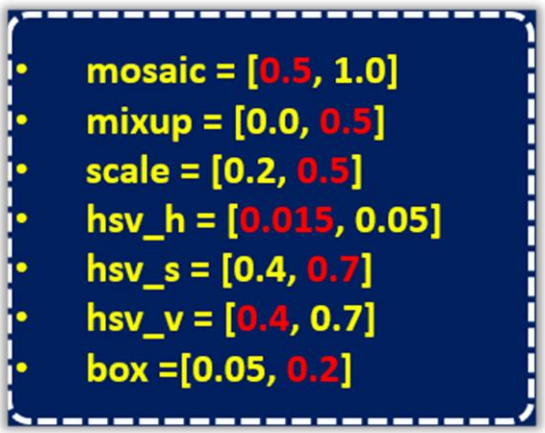


Figura 35: Los valores elegidos en la etapa 3 se han marcado en color rojo

En la tabla 7 se recoge la comparación de las métricas de los mejores modelos de cada una de estas dos etapas. En ambos casos las métricas se han obtenido para un valor del índice de confianza de 0,40 y se corresponden a la validación contra el conjunto de datos de validación.

- 50-100 épocas - conf=0,40 - Validación	Etapa 2 (v)	Etapa 3 (v)	Diff
mAP50-95	0,6412	0,6453	+0,64%
mAP50	0,8921	0,8924	+0,03%
Precision	0,8537	0,8691	+1,80%
Recall	0,8644	0,8745	+1,17%
F1 Score	0,8590	0,8718	+1,49%

Tabla 7: Comparativa de las métricas entre la etapa 2 y etapa 3 con indicación del porcentaje de mejora.

La columna "Diff" recoge las diferencias en porcentaje obtenidas para cada una de las métricas de las dos etapas. Las cinco métricas han mejorado; mejoras modestas pero positivas. La métrica mAP50-95 que indica la precisión promedio en un rango amplio de IoU, tiene una cierta mejora de 0,64%. Para mAP50, esta etapa de refinamiento ha resultado despreciable. En cambio, precision sí ha tenido un avance notable de 1,80% lo que repercutirá en la exactitud de las predicciones. Igualmente, recall ha mejorado sensiblemente, un 1,17%. Esta mejora contribuye a disminuir los falsos negativos. Estas dos últimas mejoras repercuten positivamente en el equilibrio del modelo con un incremento del 1,49% de F1 Score.

En general el refinamiento ha mejorado el modelo positivamente en especial en lo que respecta a la calidad de las predicciones.

5.2 Fase 2: entrenamiento final del modelo

Una vez que se han determinado los valores de los tres conjuntos de parámetros identificados en la etapa 1 de la anterior fase, se entrenará el modelo final.

5.2.1 Entrenamiento

Para el entrenamiento final se entrenarán 250 épocas controlando EarlyStopping estableciendo el parámetro *patien* en 100 épocas. En la figura 36 se recogen los parámetros del entrenamiento.

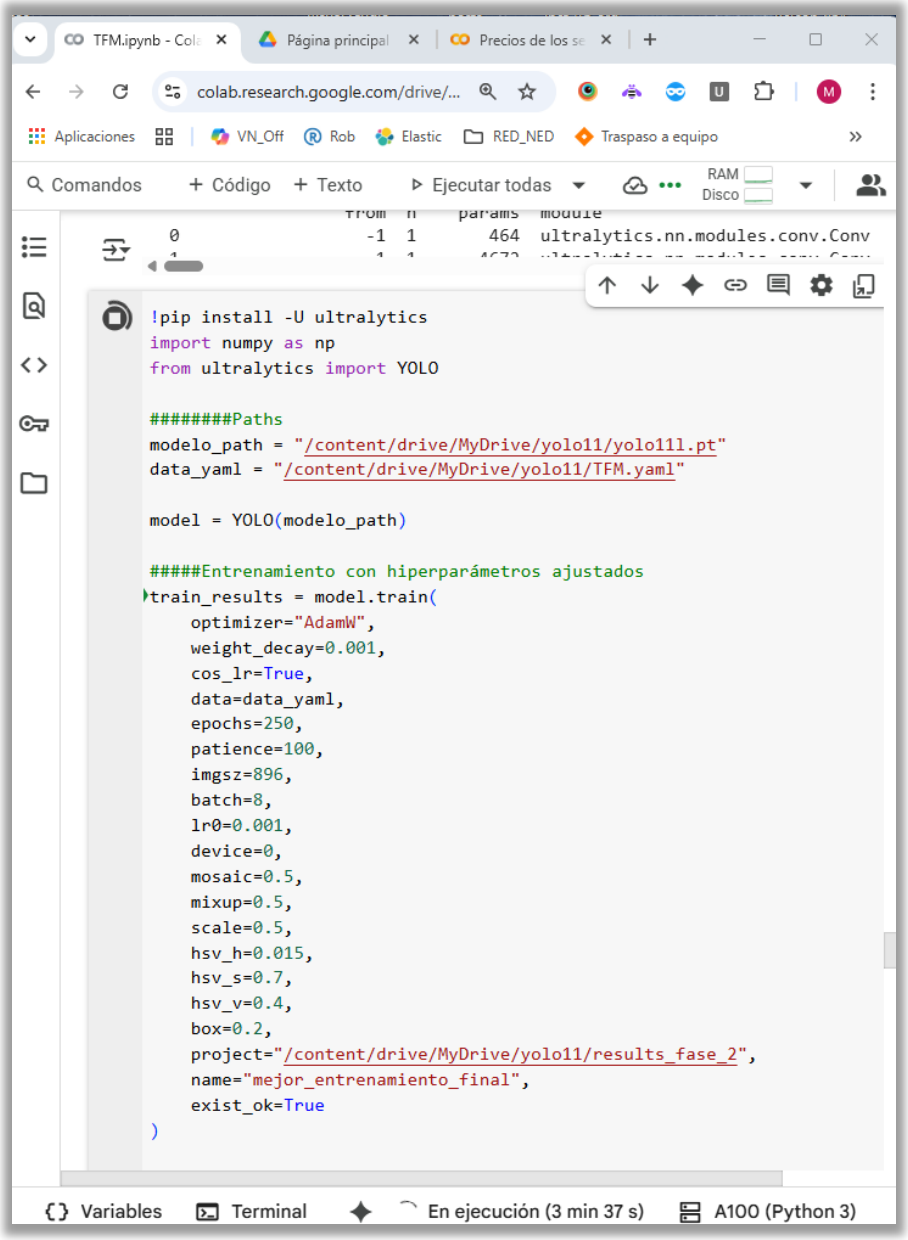


Figura 36: Entrenamiento del modelo final

5.2.1 Resultados y evaluación del modelo

A partir de la época 137 el entrenamiento no ha mostrado mejoras por lo que el hiperparámetro *patien* ha activado la parada en la época 237. Las trazas se recogen en la figura 37.

```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
    237/250      12.8G    0.02489     0.4123      1.176        188        896: 100%|██████████| 12/12 [00:02<00:00,  5.24it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 2/2 [00:00<00:00,  8.20it/s]
EarlyStopping: Training stopped early as no improvement observed in last 100 epochs. Best results observed at epoch 137, best model saved as best.pt.
To update EarlyStopping(patience=100) pass a new patience value, i.e. `patience=300` or use `patience=0` to disable EarlyStopping.


237 epochs completed in 0.253 hours.
Optimizer stripped from /content/drive/MyDrive/yolo11/results_fase_2/mejor_entrenamiento_final/weights/last.pt, 51.2MB
Optimizer stripped from /content/drive/MyDrive/yolo11/results_fase_2/mejor_entrenamiento_final/weights/best.pt, 51.2MB
```

Figura 37: EarlyStopping en el entrenamiento final

A continuación, se analizan las curvas de entrenamiento del modelo, los umbrales del índice de confianza y la matriz de confusión.

Observando las gráficas que registran el comportamiento de las métricas durante el entrenamiento recogidas en la figura 38:

metrics/precision(B): aumenta rápidamente al inicio, con ciertos desequilibrios importantes, para estabilizarse alrededor del 0,8 entre las épocas 50-100. La estabilidad sugiere que ha convergido y siendo un valor bueno el haber llegado al 0,8 (en la métrica calculada 0,87) sería interesante mejorar este valor.

metrics/recall(B): la evolución ha sido muy parecida a precision, recall sube rápidamente y se estabiliza cerca de 0,8 alrededor de las 100 épocas y con una ligera tendencia ascendente. El valor de referencia, con un índice de conf de 0,40, y validándolo sobre el conjunto de datos test, alcanza el 0,90 (figura 38), valor sólido y adecuado para empezar a probar el modelo en producción.

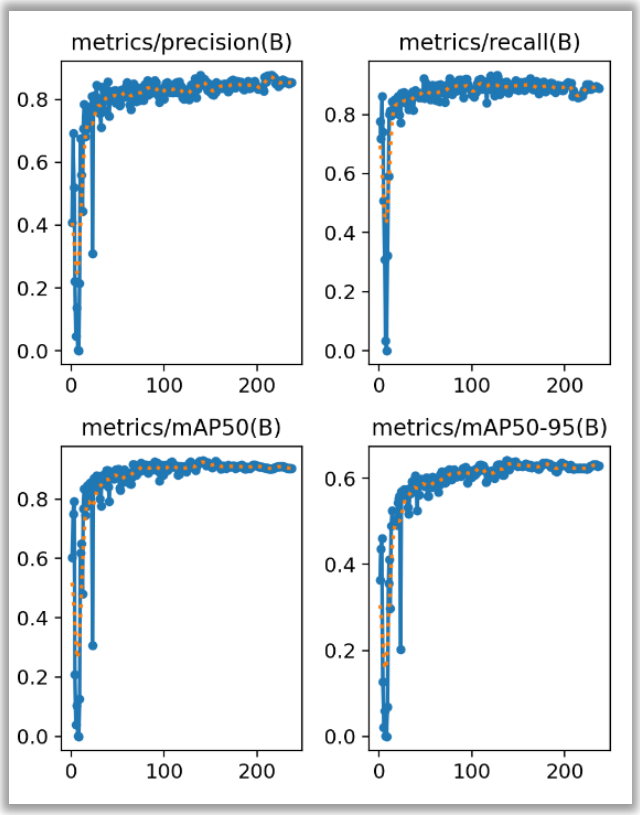


Figura 38: Gráficas de las métricas de entrenamiento

metrics/mAP50(B): esta métrica aumenta consistentemente hasta alcanzar aproximadamente 0,8 alrededor de las 75-100 épocas, con una estabilización hacia el final. Un mAP50 de 0,8 es muy bueno para un umbral IoU de 0,5 indicando una alta precisión y recuperación en detecciones menos estrictas.

metrics/mAP50-95(B): en este caso es importante tener en cuenta que en esta cuarta curva ha cambiado la escala vertical, se pinta de 0,0 a 0,6 y no a 0,8 como en las

tres gráficas anteriores. Esto revela que se estabiliza en 0,6 después de 100 épocas con una curva ascendente pero que no llega a estabilizarse completamente. La interpretación que se puede hacer es que el modelo tiene dificultades con umbrales de IoU más exigentes (0,5-0,95) infiriendo, por tanto, que todavía hay margen para buscar unas métricas más ambiciosas.

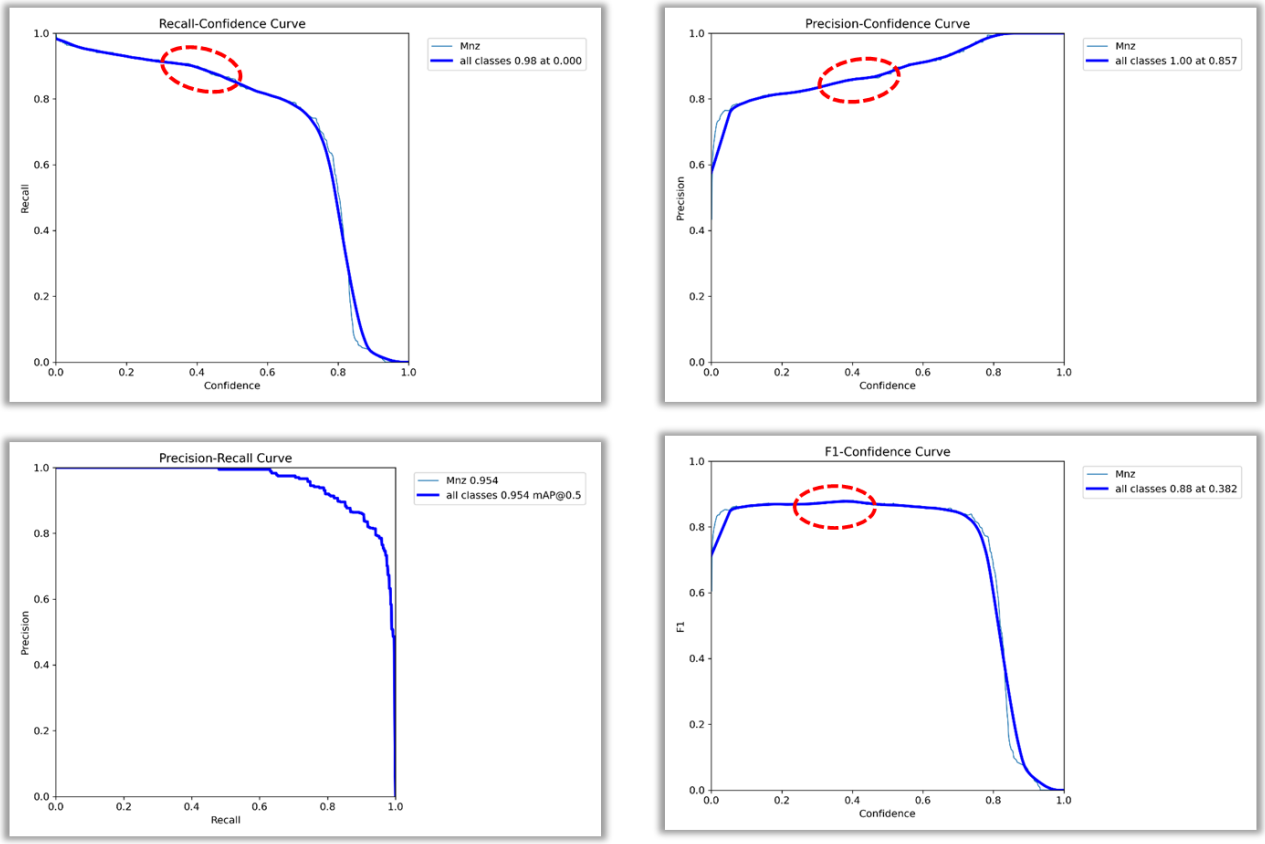


Figura 39: Umbrales de confianza vs recall, precisión y F1

Analizando las curvas de la figura 39:

Precision vs umbral del índice de confianza: la curva indica que en un umbral de confianza de 0,857 todas las predicciones son correctas, pero claro eso supone penalizar el recall (falsos negativos).

Recall vs umbral del índice de confianza: cercano a 1 para bajos umbrales de confianza, pero para equilibrar las predicciones puede ser adecuado el umbral de índice en 0,40.

Precision-Recall: esta curva está indicando un rendimiento muy alto para la clase Mnz en el caso de la métrica mAP50, el área bajo la curva (AUC) es de 0,954 reflejando un equilibrio alto entre precisión y recall.

F1 Store vs umbral del índice de confianza: según la anotación, F1 alcanza su valor máximo de 0,88 en el umbral de confianza 0,382, por lo que parece adecuado fijar el umbral de este índice en 0,40 a la hora de extraer las métricas.

Matriz de confusión: en el dataset sólo se ha identificado la clase monacita, “Mnz”, por lo que todo lo que no se reconoce como ese mineral forma parte de la clase “background”. La matriz de confusión muestra el rendimiento de un modelo de clasificación para la clase "Mnz".

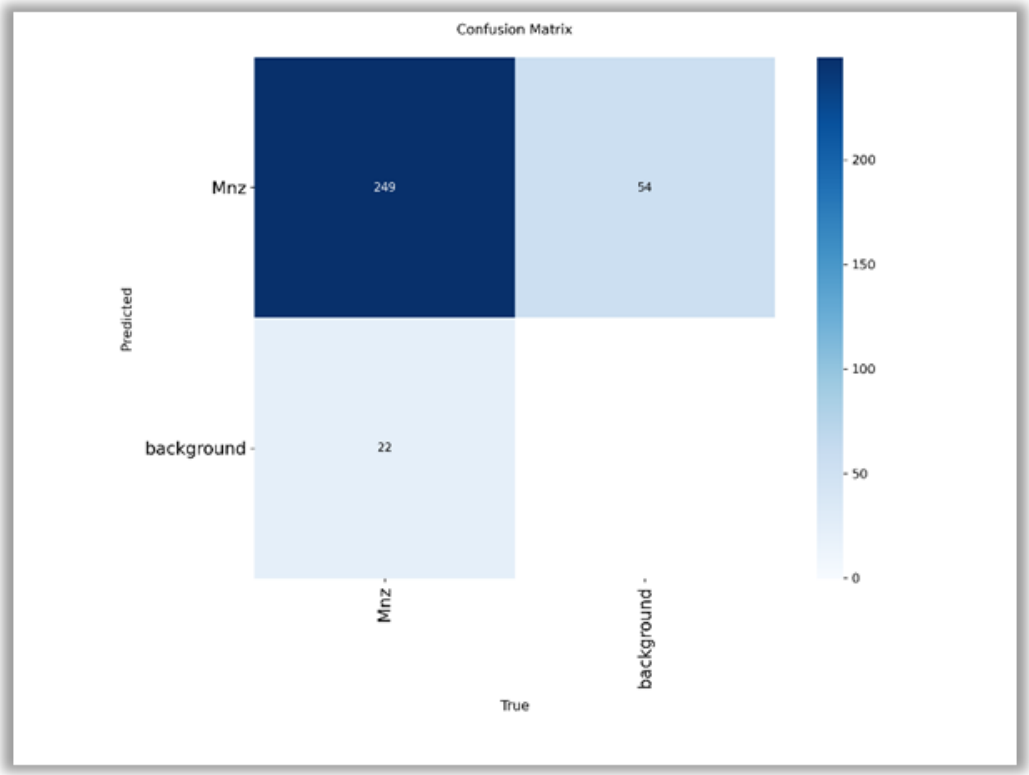


Figura 40: Matriz de confusión

- **Verdaderos positivos, TP**: el modelo predijo correctamente 249 casos como “Mnz”.
- **Falsos negativos, FN**: el modelo falló al clasificar 22 casos de “Mnz” como “background”.
- **Falsos positivos, FP**: se clasificaron 54 casos que no eran monacita como “Mnz”.

No se muestra el valor explícito para “background” por la propia naturaleza de la anotación del dataset.

Las métricas derivadas son

- **Precision** = (TP / (TP + FP)) = 249 / (249 + 54) = 0,822. El 82,2% de las predicciones de “Mnz” son predicciones correctas.
- **Recall** = (TP / (TP + FN)) = 249 / (249 + 22) = 0,919. Alta detección de los casos reales de “Mnz”, el 91,9%.
- **F1-Score** , calculando la media armónica = (2 · Precisión · Recall / (Precisión + Recall)) = 2 x 0,822 x 0,819 / (0.822 + 0.919) = 0,868, (86.8%). Buen equilibrio entre precision y recall.

Respecto de las anotaciones de las imágenes, a continuación, en la figura 41, se recogen algunos ejemplos pertenecientes al conjunto de validación test, esto es, no son predicciones sino las anotaciones reales del conjunto. En la figura 42 se muestran estos mismos ejemplos, pero ya de las predicciones que ha hecho el modelo.

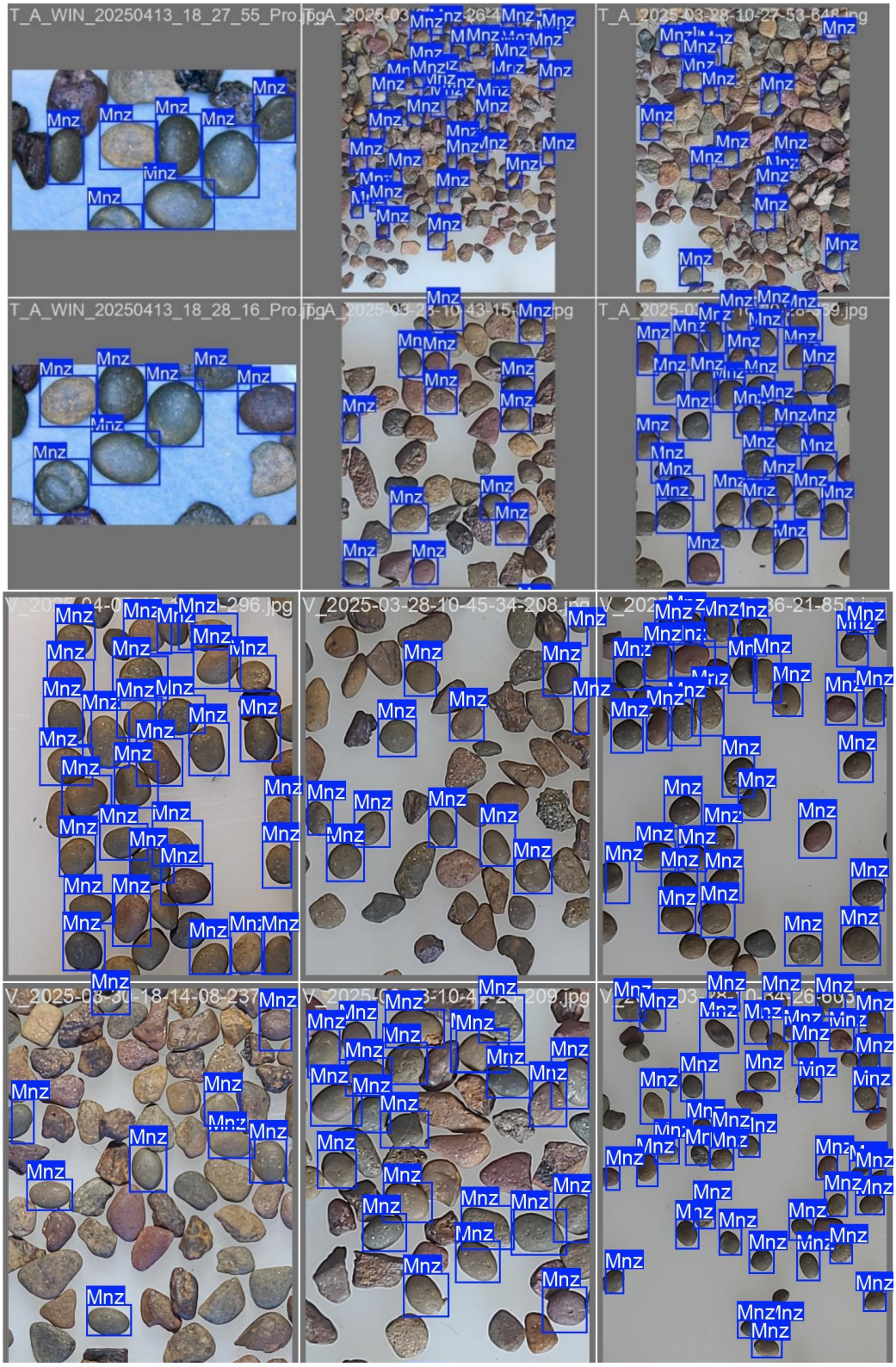


Figura 41: Ejemplos de anotaciones reales del conjunto de datos test

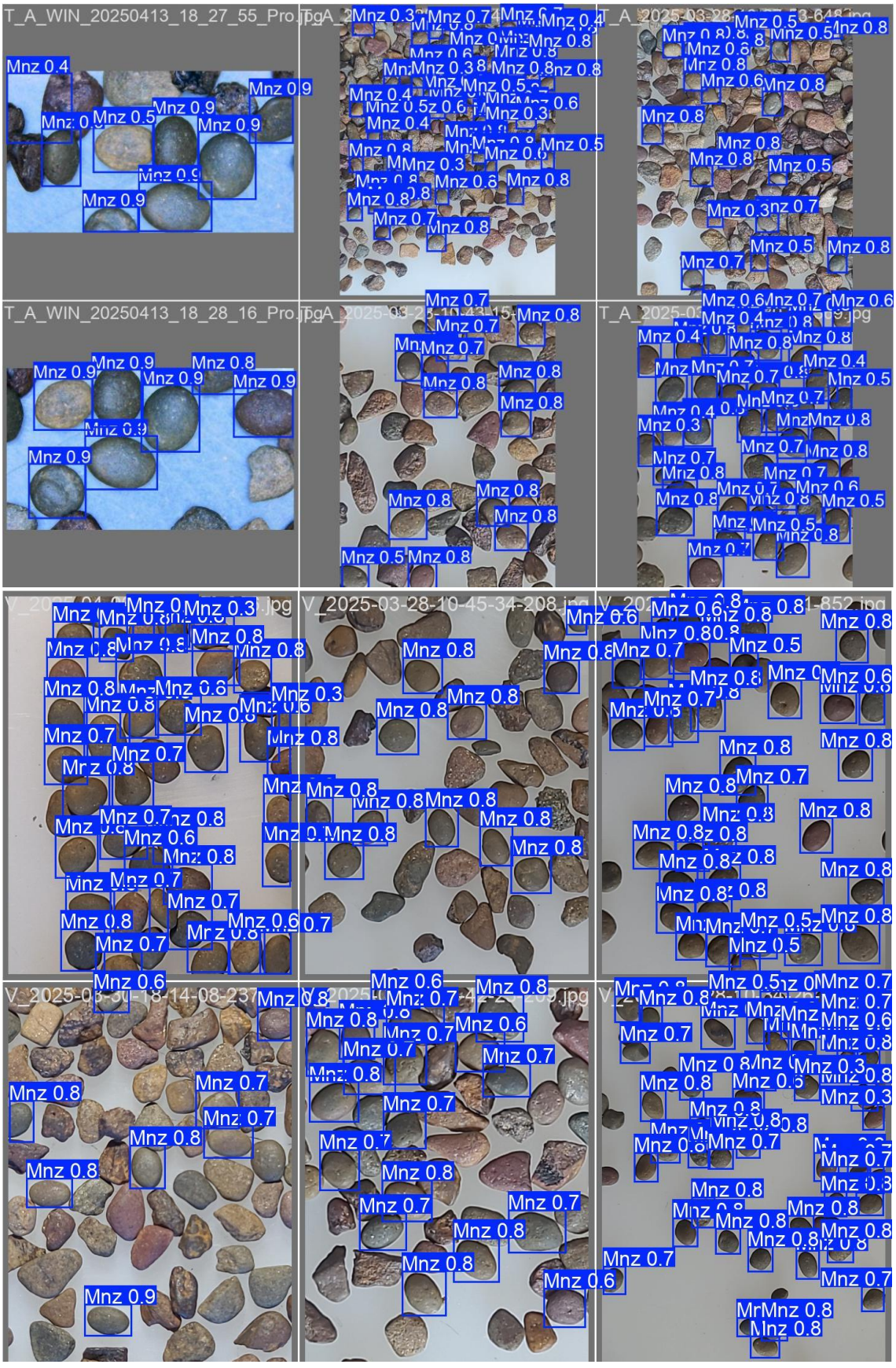


Figura 42: Ejemplos de las predicciones sobre el conjunto de datos test

El mejor modelo según mAP50-95, fijando el umbral del índice de confianza en 0,40 y habiendo sido validado contra el conjunto de datos de test, conjunto de imágenes que no han participado en ninguno de los entrenamientos anteriores de este experimento, devuelve las métricas recogidas en la figura 43.

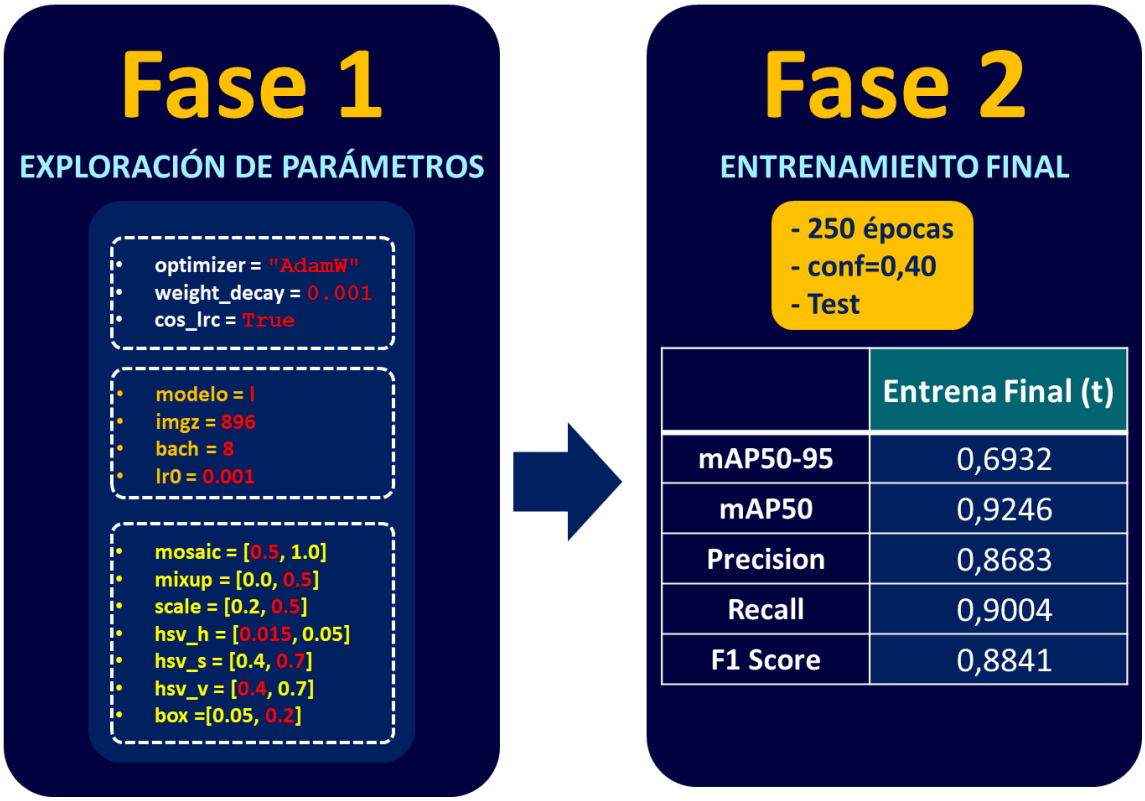


Figura 43: Esquema de las dos fases indicando los valores de las métricas del entrenamiento final

6. CONCLUSIONES Y FUTUROS TRABAJOS

6.1 Conclusiones

Sustituyendo los resultados descritos en la tabla cualitativa que se había preparado en el anterior apartado, tenemos la tabla 8:

	mAP50-95	mAP50	Precision	Recall	F1 Score	Descripción
Excelente	> 0,90	> 0,95	> 0,95	> 0,95	> 0,95	Optimización extrema, comparable al estado del arte
Muy bueno	0,75-0,9	0,9-**0,92**-0,95	0,9-0,95	0,9-**0,90**-0,95	0,9-0,95	Aplicaciones practicas, alta generalización
Bueno	0,6-**0,69**-0,75	0,7-0,9	0,8-**0,87**-0,9	0,8-0,9	0,8-**0,88**-0,9	Rendimiento sólido, propio de datasets pequeños
Aceptable	0,4-0,6	0,5-0,7	0,6-0,8	0,6-0,8	0,6-0,8	Tareas básicas. Mejorable
Bajo	< 0,4	< 0,5	< 0,5	< 0,5	< 0,5	Pobre desempeño, requiere ajustes

Tabla 8: Clasificación cualitativa del desempeño del modelo final

Gráficamente, en la tabla, se observa, que el resultado de los entrenamientos ha devuelto un **modelo Bueno** con un **gran potencial a Muy bueno.** La métrica recall, métrica que hace referencia a los falsos negativos y que para el caso de uso propuesto es la que se considera más importante por ser la que condiciona que se pasen o no por alto nódulos de monacita en las muestras comprobadas, se ubica en el rango inferior de la categoría Muy bueno, al igual que mAP50.

Hay dos fases bien diferenciadas antes de tener el conocimiento suficiente sobre un yacimiento minero que lleve a decidir explotarlo para obtener un beneficio económico: la fase de exploración y la fase de investigación. Así como existe la figura legal de la Concesión de Explotación (CE) que identifica el momento temporal y espacial en el que se desarrollan las labores de extracción del mineral para su aprovechamiento, existen dos figuras legales que acogen tanto la fase de exploración como la de investigación, estos son los derechos mineros identificados como Permiso de Exploración (PE) y Permiso de Investigación (PI). Primero se explora un territorio y en el caso de que aparezcan indicios interesantes, éstos se investigarán con detalle.

La exploración se realiza en grandes áreas frente a superficies menores en fase de investigación. Las técnicas que se emplean en cada caso también son diferentes. Igualmente, la cantidad de recursos económicos disponibles para cada una de las dos etapas poco tienen que ver. El capital, siempre dependiente del riesgo, es muy escaso en el caso de la exploración precisamente por la alta incertidumbre sobre los resultados que pueda devolver un territorio. En la fase de investigación donde, por norma general, se estudian áreas más pequeñas avaladas por exploraciones previas, es mucho más probable que los inversores se muestren interesados en invertir en el proyecto.

En el presente TFM se ha entrenado un modelo mediante una CNN capaz de identificar y cuantificar nódulos de monacita gris sobre muestras de sedimento bateadas obtenidas en la fase de exploración geológica-minera. El modelo al que se ha llegado es totalmente funcional y sirve como punto de partida para identificar de forma sistemática la concentración de cada muestreo de la fase de exploración, automatizando y estandarizando el proceso y mejorando notablemente la productividad reduciendo costes.

Dicho esto, volviendo a la tabla 8 donde se identifica muy claramente el camino recorrido y el que queda todavía por transitar, se evidencia la necesidad de mejorar especialmente la métrica mPA50-95 junto con precision.

Esto, junto con las modestas mejoras en la calidad del modelo encontradas en la etapa de refinamiento probando distintos valores para determinados hiperparámetros y, sabiendo que el dataset de entrenamiento, conjunto de datos que se ha construido para este TFM planteándolo como un MPV, es la base fundamental de cualquier modelo robusto, hace pensar en un cierto agotamiento a la hora de entrenar el modelo puesto que mPA50-95 no ha conseguido superar la barrera del 0,69. *Everingham et al. (2010)* muestra cómo errores en las anotaciones degradan la performance del modelo, por lo que parece un agotamiento debido a la calidad del dataset y posiblemente al tamaño del conjunto de datos.

Por todo ello una de las vías más prometedoras para futuros trabajos radica en la ampliación y revisión de la calidad y diversidad del conjunto de datos.

6.2 Propuestas para continuar

6.2.1 Mejora del modelo

En las conclusiones se ha evidenciado un cierto agotamiento a la hora de mejorar el modelo probablemente vinculado a ciertas limitaciones del conjunto de datos. Por ello se han identificado tres acciones para resolverlo:

1.- Ampliación del conjunto de datos: sabemos que a mayor volumen de datos mejora notablemente el comportamiento de las métricas en los entrenamientos con la consecuente mejora del desempeño del modelo, esto se ha demostrado empíricamente en concursos como ILSVRC (Imagenet Large Scale Visual Recognition Challenge) o COCO Challenge. Por ello la primera acción propuesta es la de ampliar el dataset.

Actualmente se disponen de 131 imágenes y 2.626 instancias y, atendiendo a la tabla 2 del apartado 4.2, sería interesante aumentar el número de imágenes hasta alcanzar el segundo tramo de la tabla, esto es llegar a un mínimo de 500. De mantenerse la media actual de instancias por imagen se llegaría a las 10.000 instancias.

2.- Revisión de la calidad del dataset por pares: se ha identificado la necesidad de revisar por más de una persona las anotaciones, tanto su sentido de clasificación como la precisión geométrica de las cajas delimitadoras.

Determinados conjuntos de datos de imágenes, como el del presente TFM, salvo aquellos en los que es muy obvia su clasificación, tienen una cierta componente "subjetiva" que viene dada por el conocimiento y experiencia sobre el objeto a clasificar. Como ejemplos, tipos de objetos similares al mineral que trata este trabajo u otros como pueden ser imágenes de cortes microscópicos de tejidos biológicos para identificar estructuras o características celulares determinadas, o cortes de minerales en láminas igualmente para clasificar estructuras mineralógicas, también imágenes de satélites para clasificación del suelo o cambios ambientales, pruebas diagnósticas médicas (radiografías, escáneres, resonancias, etc,) para detectar anomalías, y otras muchas. Para mitigar este grado de subjetividad es necesario establecer un protocolo claro y eficiente que vele por la calidad del dataset.

3.- Validación cruzada: para diversificar y enriquecer los entrenamientos evitando desbalanceo entre imágenes e instancias, se propone automatizar mediante un script la distribución aleatoria de los conjuntos de datos de entrenamiento y validación, siempre dejando reservado el conjunto de test. Esta distribución programática debe respetar el equilibrio entre número de imágenes y número de instancias.

Una vez resueltos estos tres puntos se debe reevaluar el modelo y obtener nuevas conclusiones.

6.2.2 Explorar modelos pequeños para obtener predicciones desde dispositivos móviles.

En el apartado 5.1.1.3.2, se ha indicado que el tamaño del modelo preentrenado influye en el tamaño del modelo final y a medida que se incrementa el número de variables crecen igualmente las necesidades computacionales requeridas para ejecutarlos, por ello se justifica evaluar modelos que, aun sacrificando cierta exactitud, permitan una mayor versatilidad en contextos donde los recursos de hardware son limitados. De ahí la presente propuesta de explorar y estudiar el mejor modelo preentrenado con nano (yolo11n.pt) para procurar la movilidad de las predicciones.

6.2.3 Tantear dataset de muestras húmedas.

Resulta pertinente, y podría abrir nuevas vías de trabajo, explorar el comportamiento y las posibilidades de entrenar un modelo con un dataset compuesto por muestras húmedas. Hay unas diferencias sustanciales entre las muestras secas y las húmedas tanto en la representación visual por la variación de brillos y tonalidades como por su comportamiento físico; la muestra se apelmaza porque la humedad tiende a agrupar las fracciones más finas impidiendo conformar una muestra homogénea desde el punto de vista granulométrico.

BIBLIOGRAFÍA Y RECURSOS DE INTERNET

→ **Adamas Intelligence.** (s.f). http://www.adamasintel.com/

→ **Adamas Intelligence**. (2024). Global rare earth element production grew slightly to 350,000 tonnes in 2023, expected to rebound strongly in 2024. Green Car Congress. https://www.greencarcongress.com/2024/09/20240902-adamas.html

→ (*) **Andrej Karpathy**. (s.f) Wikipedia. https://es.wikipedia.org/wiki/Andrej_Karpathy

→ (*) **Cheng, Baolian & Bradley, P**. (2023). What Machine Learning Can and Cannot Do for Inertial Confinement Fusion. Plasma. 6. 334-344. 10.3390/plasma6020023.

→ (**) **Díaz Soto, J. L**. (2019). Deep learning: Redes neuronales convolucionales aplicadas a la clasificación de Viena. TFM, Universidad Complutense de Madrid. https://docta.ucm.es/entities/publication/5545d014-6394-46da-a512-941cd88ad9fb

→ (**) **Ding, J., Xue, N., Long, Y., Xia, G. S., & Lu, Q**. (2018). Learning RoI Transformer for Oriented Object Detection in Aerial Images.CVPR 2019 (OBB-DOTA dataset). https://arxiv.org/pdf/1812.00155

→ **Donnot, M., Guigues, J., Lulzac, Y. et al.** Un nouveau type de gisement d'europium: la monazite grise à europium en nodules dans les schistes paléozoïques de Bretagne. Mineral. Deposita 8, 7–18 (1973). https://doi.org/10.1007/BF00203346

→ **EFE Servicios**. Tierras raras [Noticia multimedia]. EFS. (2025). https://efs.efeservicios.com/noticia-multimedia/tierras-raras/55015920597

→ **EURARE Project**. EURARE - Developing Europe's rare earth element deposits. https://www.eurare.org/

→ (**) **Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A**. (2010). *The Pascal Visual Object Classes (VOC) Challenge***.** International Journal of Computer Vision, 88(2), 303–338. https://doi.org/10.1007/s11263-009-0275-4

→ **Fan, C., Liu, Y., & Zhou, J.** (2021). Deep learning of rock images for intelligent lithology identification. Computers & Geosciences, 152, Article S009830042100100X. https://doi.org/10.1016/j.cageo.2021.104636

→ (*) **Frazier, Amy & Song, Lei**. (2024). Artificial Intelligence in Landscape Ecology: Recent Advances, Perspectives, and Opportunities. Current Landscape Ecology Reports. 10. 10.1007/s40823-024-00103-7.https://www.researchgate.net/publication/385863914_Artificial_Intelligence_in_Landscape_Ecology_Recent_Advances_Perspectives_and_Opportunities

→ **Fridman, L.** (2017). MIT 6.S094: Convolutional Neural Networks for End-to-End Learning of the Driving Task [Video]. YouTube. https://www.youtube.com/watch?v=U1toUkZw6VI

→ **Fridman, L**. (2017). MIT 6.S094: Computer Vision [Video]. YouTube. https://www.youtube.com/playlist?list=PLrAXtmErZgOeiKm4sgNOknGvNjby9efdf

→ **Fridman, L.** (2017, October 15). MIT 6.S094: Deep Learning Basics: Introduction and Overview [Video]. YouTube. https://www.youtube.com/watch?v=O5xeyoRL95U

→ (*)(**) **Fridman, L**. (2020, January 13). Deep Learning State of the Art (2020) [Video]. YouTube. https://www.youtube.com/watch?v=0VH1Lim8gL8&list=PLrAXtmErZgOeiKm4sgNOknGvNjby9efdf

→ **Flores Vidal, P. A**. (2019). Problemas de tratamiento de imágenes basados en la incorporación de características humanas. https://docta.ucm.es/entities/publication/53584115-ffc3-4577-b8a4-dd4ce3a28d5e

→ **Flores Vidal, P., Castro, J., & Gómez, D**. (2022). Postprocessing of edge detection algorithms with machine learning techniques. Computational Intelligence and Neuroscience, 2022, Article ID 9729343. https://doi.org/10.1155/2022/9729343

→ **García-Arias, S., & Velandia Patiño, F. A**. (2025). Modelos de redes neuronales convolucionales como herramienta para automatizar la clasificación de rocas. Revista EIA, 22(43), 4317, 1–28. https://doi.org/10.24050/reia.v22i43.1813

→ **García Martínez, J. (2024**). Reconocimiento visual de aves con Deep Learning (Trabajo Fin de Grado). Universidad de Alicante. https://rua.ua.es/dspace/handle/10045/145620

→ **Gil Ibarguchi. José Ignacio**. Prospección y estudio de yacimientos de monacita gris en España. Tierra y Tecnología. Instituto de Ciencias de la Geología. https://www.icog.es/TyT/index.php/2025/04/prospeccion-y-estudio-de-yacimientos-de-monacita-gris-en-espana/

→ **Hacıefendioğlu, K., Adanur, S. & Demir, G.** Automatic Landslide Segmentation Using a Combination of Grad-CAM Visualization and K-Means Clustering Techniques. Iran J Sci Technol Trans Civ Eng 48, 943–959 (2024). https://doi.org/10.1007/s40996-023-01193-9

→ **Herrera, G**, C.-D. (2024, December). **CSIC** Investiga 8. Revista de Ciencia. Consejo Superior de Investigaciones Científicas. Pag 62-34. (España). http://doi.org/10.20350/DIGITALCSIC/17155

→ **Huang, Yiyuan**. (2024). Aplicación de modelos YOLO-NAS y YOLOv8 en sistemas seguimiento de múltiples objetos. Master, E.T.S. de Ingenieros Informáticos (UPM). https://oa.upm.es/view/institution/ETSI=5FInformatica/

→ **Inglés Muñoz, G**. (2023). Técnicas de aprendizaje profundo para la detección automática de objetos en imágenes sonar (Proyecto Fin de Grado). Universidad Politécnica de Cartagena.https://repositorio.upct.es/entities/publication/f8c08fe3-d73c-4900-8833-252ac459a6c5

→ **Jegham, Nidhal & Koh, Chan Young & Abdelatti, Marwan & Hendawi, Abdeltawab.** (2024). Evaluating the Evolution of YOLO (You Only Look Once) Models: A

Comprehensive Benchmark Study of YOLO11 and Its Predecessors. 10.48550/arXiv.2411.00201. https://doi.org/10.48550/arXiv.2410.17725

→ **Junayed, Masum Shah & Jeny, Afsana & Islam, Md Baharul & Ahmed, Ikhtiar & Shah, A. F. M. Shahen.** (2022). An Efficient End-to-End Deep Neural Network for Interstitial Lung Disease Recognition and Classification. 10.48550/arXiv.2204.09909. https://www.researchgate.net/publication/360098647_An_Efficient_End-to-End_Deep_Neural_Network_for_Interstitial_Lung_Disease_Recognition_and_Classification

→ (**) **Krizhevsky, A., Sutskever, I., & Hinton, G. E**. (2012). ImageNet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, 25, 1097–1105. https://papers.nips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

→ (*)(**) **LeCun, Y., Bengio, Y., & Hinton, G**. (2015). Deep Learning. Nature, 521(7553), 436-444. https://www.nature.com/articles/nature14539

→ (*) **Lemenkova, Polina**. (2024). Deep Learning Methods of Satellite Image Processing for Monitoring of Flood Dynamics in the Ganges Delta, Bangladesh. Water. 16. 1141. 10.3390/w16081141.

→ (**) **Lin, TY. et al.** (2014). Microsoft COCO: Common Objects in Context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8693. Springer, Cham. https://doi.org/10.1007/978-3-319-10602-1_48

→ (*) **Mahajan, Abhishek & Chakrabarty, Nivedita & Majithia, Jinita & Ahuja, Ankita & Agarwal, Ujjwal & Suryavanshi, Shubham & Biradar, Mahesh & Sharma, Prerit & Raghavan, Bagyam & Arafath, Rasheed & Shukla, Shreya**. (2023). Multisystem Imaging Recommendations/Guidelines: In the Pursuit of Precision Oncology. Indian Journal of Medical and Paediatric Oncology. 44. 002-025. 10.1055/s-0043-1761266.

https://www.researchgate.net/publication/369034471_Multisystem_Imaging_RecommendationsGuidelines_In_the_Pursuit_of_Precision_Oncology

→ **Parlamento Europeo y Consejo de la Unión Europea.** (2024). Reglamento (UE) 2024/1252 del Parlamento Europeo y del Consejo de 11 de abril de 2024 por el que se establece un marco para garantizar un suministro seguro y sostenible de materias primas fundamentales y por el que se modifican los Reglamentos (UE) n.º 168/2013, (UE) 2018/858, (UE) 2018/1724 y (UE) 2019/1020**.** Diario Oficial de la Unión Europea, https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=CELEX%3A32024R1252

→ **Rasheed, A. F., & Zarkoosh, M.** (2024). Yolov11 optimization for efficient resource utilization. arXiv. https://doi.org/10.48550/arXiv.2412.14790

→ **Regueiro, M.** (2019). ¿Qué son las tierras raras? Tierra y Tecnología – Ilustre Colegio Oficial de Geólogos. https://www.icog.es/TyT/index.php/2019/05/que-son-las-tierras-raras/

→ **Rodríguez Escabias, Daniel**. (2023). Aplicación de técnicas de visión por computador para medir el riesgo de contagio por virus en aeropuertos. Master, E.T.S. de Ingenieros Informáticos (UPM). https://oa.upm.es/75308/3/TFM_DANIEL_RODRIGUEZ_ESCA.pdf

→ **Redmon, J., Divvala, S., Girshick, R., & Farhadi, A**. (2016). You only look once: Unified, real-time object detection. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 779-788. https://arxiv.org/abs/1506.02640

→ **Rosenblum, Sam, and Elwin L. Mosier.** (1983). Mineralogy and Occurrence of Europium-Rich Dark Monazite. Geological Survey Professional Paper 1181. Washington, DC: U.S. Geological Survey. https://pubs.usgs.gov/pp/1181/report.pdf

→ **Swanson, A.** (2025). China tightens grip on samarium, a rare earth vital to fighter jets. The New York Times. https://www.nytimes.com/2025/06/09/business/china-rare-earth-samarium-fighter-jets.html

→ **U.S. Geological Survey**. (2025). Mineral Commodity Summaries 2025 (vers. 1.2, 3 de marzo de 2025). U.S. Geological Survey. https://doi.org/10.3133/mcs2025

→ **Ultralytics.** (2024). YOLOv11 - Ultralytics. Retrieved from https://github.com/ultralytics/ultralytics Licensed under AGPL-3.0: https://www.gnu.org/licenses/agpl-3.0.html

→ **Ultralytics.** (2023, 27 de noviembre). *The importance of high-quality computer vision datasets***.** Ultralytics. https://www.ultralytics.com/blog/the-importance-of-high-quality-computer-vision-datasets

→ **Ultralytics.** (s.f.). Recopilación y anotación de datos. Ultralytics. https://docs.ultralytics.com/es/guides/data-collection-and-annotation/#how-can-i-ensure-high-consistency-and-accuracy-in-data-annotation

→ **Vaquero Nazabal, C.** (1979). Descubrimiento monacita de facies aberrante. Boletín Geológico y Minero, XC-IV, 374–379. https://info.igme.es/biblioteca/ficheros/BGM/Boletin%2090_4_1979.pdf

→ (**) **Vaswani, A., Shazeer, N., Parmar, N., et al**. (2017). Attention is All You Need. arXiv preprint arXiv:1706.03762. https://arxiv.org/abs/1706.03762

→ **Zhao, Tianjie et al.** (2024). Artificial intelligence for geoscience: Progress, challenges, and perspectives. The Innovation, 5(5), 100691. https://www.sciencedirect.com/science/article/pii/S2666675824001292

→ (**) **Zhou, X., Wang, D., & Krähenbühl, P**. (2019). Objects as Points (CenterNet). https://arxiv.org/pdf/1904.07850