



universidad
de león

Departamento de Ingenierías
Mecánica, Informática y Aeroespacial

MÁSTER UNIVERSITARIO EN ROBÓTICA E INTELIGENCIA ARTIFICIAL

Trabajo de Fin de Máster

Detección de células redondas en un conjunto de
imágenes médicas mediante aprendizaje profundo

Round Cell Detection in Medical Image Sets Using
Deep Learning

Autor: Aitor García Blanco

Tutor: Laura Fernández Robles

UNIVERSIDAD DE LEÓN
Departamento de Ingenierías Mecánica, Informática y Aeroespacial
MÁSTER UNIVERSITARIO EN ROBÓTICA E INTELIGENCIA ARTIFICIAL
Trabajo de Fin de Máster

ALUMNO: Aitor García Blanco

TUTOR: Laura Fernández Robles

TÍTULO: Detección de células redondas en un conjunto de imágenes médicas mediante aprendizaje profundo

TITLE: Round Cell Detection in Medical Image Sets Using Deep Learning

CONVOCATORIA: Septiembre, 2025

RESUMEN: Con el objetivo de optimizar el análisis seminal, un proceso clave en la medicina reproductiva, se aborda el diseño, desarrollo y validación de un sistema automatizado para la detección de células redondas en imágenes de muestras de semen humano sin procesar, mediante técnicas de aprendizaje profundo y visión por computador.

La metodología se centra en la consolidación de un dataset validado por expertos. Posteriormente, se aborda el entrenamiento y la optimización de diversas arquitecturas del modelo YOLO (de la v8 a la v12), incluyendo la creación de un modelo con arquitectura personalizada y otro ensamblado (*ensemble*) que fusiona las fortalezas de YOLOv10s y YOLOv12s. Para maximizar su rendimiento se aplican técnicas de ajuste fino (*fine-tuning*) y aumento de datos (*data augmentation*), junto con una sistemática optimización de hiperparámetros con Optuna y una evaluación mediante validación cruzada. Los resultados demuestran que el mejor modelo para abordar esta problemática es el YOLOv12s, que supera significativamente el rendimiento de sistemas anteriores.

El proyecto culmina con el desarrollo de una herramienta web interactiva que traduce los diferentes modelos de inteligencia artificial en una solución práctica y funcional, que facilita la validación de resultados por parte del especialistas y sienta las bases para su futura integración en entornos clínicos, potenciando así la transformación digital en el campo de la andrología.

ABSTRACT: To optimize semen analysis, a key process in reproductive medicine, this project addresses the design, development, and validation of an automated system for detecting round cells in images of unprocessed human semen samples using deep learning and computer vision techniques.

The methodology focused on the consolidation of an expert-validated dataset. Subsequently, various YOLO model architectures (from v8 to v12) were trained and optimized, including the creation of a custom architecture model and an ensemble model that combines the strengths of YOLOv10s and YOLOv12s. To maximize performance, techniques such as fine-tuning and data augmentation were applied, along with systematic hyperparameter optimization using Optuna and evaluation through cross-validation. The results demonstrate that the best model to address this problem is YOLOv12s, which significantly outperforms previous systems.

The project culminates in the development of an interactive web tool that translates the artificial intelligence models into a practical and functional solution. This tool facilitates the validation of results by specialists and lays the groundwork for its future integration into clinical environments, thereby advancing the digital transformation in the field of andrology.

Palabras clave: detección de objetos (*object detection*), células redondas (*round cells*), análisis de semen (*semen analysis*), inteligencia artificial (*artificial intelligence*), aprendizaje profundo (*deep learning*), visión por computador (*computer vision*), YOLO

Índice

Índice de figuras	IV
Índice de tablas	VI
Glosario de términos	VII
1. Introducción y objetivos	1
1.1. Introducción	1
1.2. Objetivos	2
2. Antecedentes	3
2.1. Contexto y motivación	3
2.2. Estado del arte e hipótesis	4
2.2.1. Visión por computador y modelos de detección de objetos . .	5
2.2.2. Técnicas de optimización y validación de modelos	8
2.2.3. Literatura de la investigación	9
2.2.4. Hipótesis de trabajo	10
2.3. Definición del problema	11
3. Gestión de proyecto <i>software</i>	14
3.1. Alcance del proyecto	14
3.1.1. Definición del proyecto	14
3.1.2. Presupuesto	14
3.2. Plan de trabajo	17
3.2.1. Metodología	17
3.2.2. Identificación de tareas	18
3.2.3. Estimación de tareas	20
3.2.4. Planificación de tareas	21
3.3. Gestión de recursos	22
3.3.1. Especificación y asignación de recursos	22
3.4. Gestión de riesgos	23
3.4.1. Identificación y análisis de riesgos	23
3.5. Legislación y normativa	25

4. Metodología	27
4.1. Gestión y preprocesado del <i>dataset</i>	27
4.2. Diseño experimental, entrenamiento y evaluación de modelos.	28
4.3. Desarrollo y despliegue de la herramienta web	29
5. Experimentación	30
5.1. <i>dataset</i>	30
5.2. Entorno de desarrollo	31
5.2.1. <i>Hardware</i>	31
5.2.2. <i>software y Frameworks</i>	32
5.3. Configuraciones	33
5.3.1. Entrenamiento	33
5.3.2. Validación cruzada	39
5.3.3. Ensamblado de modelos	41
5.3.4. Modelo personalizado	42
5.4. Métricas	43
6. Resultados	45
6.1. Resultados cuantitativos	45
6.2. Resultados cualitativos	48
7. Herramienta Web	50
7.1. Objetivos de la herramienta	50
7.2. Arquitectura	50
7.2.1. Organización del proyecto	51
7.2.2. Capas funcionales en <i>Streamlit</i>	51
7.2.3. Flujo de ejecución	52
8. Conclusión	53
8.1. Conclusión general	53
8.2. Limitaciones del estudio	54
8.3. Lineas de trabajo futuro	55
Bibliografía	56
A. Control de versiones	59
B. Seguimiento del proyecto	60
C. Herramienta Web	61

C.1. Manual de usuario	61
C.2. Recomendaciones de uso	64
C.3. Resolución de problemas	64
D. Evaluación comparativa de arquitecturas	66
D.1. Matrices de confusión	67
D.2. Curvas F1-confianza	71
D.3. Resultados cualitativos	75

Índice de figuras

2.1.	Diagramas de arquitectura de YOLOv8, YOLOv9 y YOLOv10	6
2.2.	Diagramas de arquitectura de YOLOv11 y YOLOv12	7
2.3.	Muestra de semen humano sin anotaciones.	11
2.4.	Ilustración con anillos de Newton.	12
2.5.	Ilustración de anotaciones para <i>feedback</i> con experto.	13
5.1.	mAP@0.5:0.95 durante el proceso de entrenamiento.	36
5.2.	mAP@0.5 durante el proceso de entrenamiento.	36
5.3.	<i>Precision</i> del modelo sobre el conjunto de validación.	37
5.4.	<i>Recall</i> del modelo sobre el conjunto de validación.	37
5.5.	<i>Box Loss</i> sobre el conjunto de entrenamiento y validación.	38
5.6.	<i>Classification Loss</i> sobre el conjunto de entrenamiento y validación. .	38
5.7.	Representación de la validación cruzada (k-fold) para todos los modelos.	40
B.1.	Diagrama de Gantt para la planificación del proyecto	60
C.1.	Interfaz de la aplicación web.	61
C.2.	Herramienta web: <i>checkbox ground truth</i>	62
C.3.	Herramienta web: <i>checkbox zoom</i>	62
C.4.	Herramienta web para procesar un conjunto de imágenes.	63
C.5.	Herramienta web: métricas y matriz de confusión.	63
D.1.	Matrices de confusión para los modelos evaluados en original_test .	67
D.2.	Matrices de confusión para los modelos evaluados en test	68
D.3.	Matrices de confusión para los modelos evaluados en test2	69
D.4.	Matrices de confusión para los modelos evaluados en test3	70
D.5.	Curvas F1-confianza para los modelos evaluados en original_test . .	71
D.6.	Curvas F1-confianza para los modelos evaluados en test	72
D.7.	Curvas F1-confianza para los modelos evaluados en test2	73
D.8.	Curvas F1-confianza para los modelos evaluados en test3	74
D.9.	Resultados comparativos para la Figura 7	75
D.10.	Resultados comparativos para la Figura 9	76
D.11.	Resultados comparativos para la Figura 38	77
D.12.	Resultados comparativos para la Figura 95	78

D.13.Resultados comparativos para la Figura 139	79
D.14.Resultados comparativos para la Figura 142	80
D.15.Resultados comparativos para la Figura 221	81
D.16.Resultados comparativos para la Figura 347	82
D.17.Resultados comparativos para la Figura 415	83
D.18.Resultados comparativos para la Figura 461	84

Índice de tablas

3.1.	Presupuesto para los recursos humanos	15
3.2.	Desglose de costes de <i>hardware</i>	16
3.3.	Presupuesto para los recursos de <i>hardware</i>	16
3.4.	Presupuesto final del proyecto	17
3.5.	Historia de usuario 1 (Planificación, configuración e investigación) . .	18
3.6.	Historia de usuario 2 (Preprocesamiento de datos y modelo base) . .	18
3.7.	Historia de usuario 3 (Entrenamiento y optimización)	19
3.8.	Historia de usuario 4 (Modelos y evaluación comparativa)	19
3.9.	Historia de usuario 5 (Desarrollo de la herramienta web interactiva) .	19
3.10.	Historia de usuario 6 (Documentación final y entrega)	19
3.11.	Estimación de tareas del proyecto	20
3.12.	Ánálisis de riesgos del proyecto	24
5.1.	Distribución del <i>dataset</i> original	30
5.2.	Resumen del <i>dataset</i> de actuación	31
5.3.	Parámetros principales de entrenamiento para cada modelo	34
5.4.	Principales hiperparámetros óptimos seleccionados para cada modelo	34
5.5.	Hiperparámetros de <i>data augmentation</i>	35
5.6.	Resultados de la validación cruzada (<i>k-fold</i>) para todos los modelos .	39
5.7.	Estructura del <i>backbone</i> del modelo personalizado	42
5.8.	Estructura del <i>head</i> del modelo personalizado	42
6.1.	Resultados de los modelos en la prueba de concepto inicial	45
6.2.	Resultados de los modelos sobre el conjunto de <i>test</i> original	45
6.3.	Evaluación de modelos sobre los nuevos conjuntos de <i>test</i>	47
6.4.	Evaluación de modelos sobre imágenes con artefactos	48

Glosario de términos

Bounding box: Rectángulo que delimita la posición y tamaño de un objeto en una imagen.

Dataset: Conjunto de datos estructurados, utilizado para entrenar, validar o evaluar modelos de inteligencia artificial.

IoU (*Intersection over Union*): Área de unión entre dos *bounding boxes* (anotación y predicción).

Métricas: Indicadores cuantitativos para la evaluación de los modelos de inteligencia artificial.

Experto: Persona con conocimientos especializados en un dominio específico (como biólogos, biotecnólogos, patólogos, etc.).

Detección de objetos: Tarea de localizar y clasificar instancias de objetos en imágenes mediante *bounding boxes* y etiquetas.

NMS (*Non-Maximum Suppression*): Algoritmo para eliminar las detecciones redundantes, manteniendo la caja con mayor confianza entre las solapadas.

Aprendizaje profundo o *deep learning*: Área del aprendizaje automático que emplea redes neuronales profundas para aprender representaciones y resolver tareas complejas.

Aprendizaje por transferencia (*Transfer Learning*): Estrategia de aprendizaje automático que consiste en reutilizar un modelo preentrenado en una tarea para aplicarlo como punto de partida en una segunda tarea distinta pero relacionada. En lugar de entrenar un modelo desde cero, se aprovecha el "conocimiento" (pesos y características) ya adquirido.

Ajuste fino (*Fine-Tuning*): Técnica específica del aprendizaje por transferencia. Consiste en tomar un modelo preentrenado y continuar su entrenamiento sobre un nuevo conjunto de datos más específico. Durante este proceso, los pesos del modelo se "ajustan" o refinan, generalmente con una tasa de aprendizaje baja, para adaptar las características aprendidas previamente a las demandas de la nueva tarea.

Framework: Conjunto estructurado de herramientas, librerías y convenciones que facilita y agiliza el desarrollo de aplicaciones o sistemas.

Streamlit: Framework de Python para crear aplicaciones web interactivas orientadas a la ciencia de datos y *Machine Learning*.

Monolítica: Arquitectura de software en la que la aplicación se despliega y ejecuta en un mismo proceso.

LabelImg: Herramienta gráfica de código abierto para la anotación de imágenes con *bounding boxes*.

Widget: Elemento de una interfaz interactiva que permite al usuario controlar el comportamiento de una aplicación.

Ground truth: Hace alusión a las anotaciones correctas que describen las *bounding boxes* de las imágenes.

Checkbox: Es un *widget* que permite al usuario activar o desactivar una opción (booleano).

Linter: Herramienta que analiza automáticamente el código fuente para detectar errores de sintaxis u otros posibles fallos antes de ejecutar el programa.

Células redondas: Según la OMS [1], toda célula no espermática que se encuentra en el eyaculado. Agrupa principalmente dos tipos: células germinales inmaduras y leucocitos.

Células germinales inmaduras: Células precursoras de los espermatozoides que no han completado su desarrollo.

Leucocitos: Glóbulos blancos, principalmente neutrófilos.

Espermograma: Consiste en analizar los espermatozoides de un hombre o un animal para evaluar la fertilidad y detectar posibles anomalías.

Espermatogénesis: Proceso biológico de formación y producción de espermatozoides.

Andrología: Área de la medicina que se centra en el estudio e investigación de cualquier aspecto relacionado con la función sexual y reproducción masculina (Por ejemplo: infertilidad masculina).

Anillos de Newton: Interferencia de la luz cuyo resultado es un patrón de anillos muy característico.

Overfitting: Proceso en el que un modelo de aprendizaje automático aprende demasiado bien los datos de entrenamiento, perdiendo capacidad de generalización y obteniendo peores resultados en las pruebas de evaluación.

TP (Verdaderos Positivos): Células redondas correctamente detectadas por el modelo. El algoritmo identifica correctamente una célula redonda, que efectivamente, existe en la muestra de semen.

FP (Falsos Positivos): Detecciones incorrectas. Donde el modelo identifica una célula redonda que en realidad no existe o corresponde a otro elemento (como un espermatozoide sin cola o un artefacto visual).

TN (Verdaderos Negativos): Regiones de la imagen correctamente no marcadas como células redondas. Representan áreas donde el modelo no detecta ninguna célula.

FN (Falsos Negativos): Células redondas presentes en la muestra que el modelo no detecta.

GVIS: Group for Vision and Intelligent Systems de la Universidad de León.

Inferencia: Proceso de utilizar un modelo entrenado para hacer predicciones sobre imágenes nuevas o no vistas previamente.

PASCAL VOC: Formato de anotación de imágenes (*Pattern Analysis, Statistical Modelling and Computational Learning Visual Object Classes*). Es un estándar para la anotación de objetos en imágenes mediante *bounding boxes* rectangulares, almacenadas en archivos XML. Cada anotación incluye las coordenadas del rectángulo (xmin, ymin, xmax, ymax) y la clase del objeto.

COCO: Formato de anotación (*Common Objects in Context*) que proporciona un estándar más completo para la detección de objetos, segmentación y etiquetado de imágenes. Utiliza archivos JSON para almacenar información sobre imágenes, categorías y anotaciones.

1. Introducción y objetivos

1.1. Introducción

El presente Trabajo de Fin de Máster, desarrollado en colaboración con la empresa Microptic S.L [2], compañía de Hamilton Thorne [3], parte del desarrollo de una Prueba de Concepto (*Proof of Concept*) desarrollada en el marco europeo DIGIS3 [4] para la detección de células redondas en imágenes médicas mediante inteligencia artificial.

El objetivo de esta solución es la identificación automatizada de células redondas en imágenes de muestras de semen humano. Esta capacidad es de gran interés para Microptic [2], ya que optimiza los tiempos de análisis de los expertos y enriquece el estudio de las muestras espermáticas, mejorando la caracterización de parámetros clave tanto en el ámbito de la reproducción asistida humana como en la investigación y producción animal.

La metodología se ha centrado, en primer lugar, en la consolidación de un conjunto de datos de alta calidad. Para ello, se procesó y validó el *feedback* proporcionado por los expertos de Microptic [2], refinando el etiquetado de las imágenes que constituyen la base para la evaluación de los modelos.

Posteriormente, aplicando técnicas de aprendizaje profundo (*Deep Learning*) y visión por computador, se adaptaron y reentrenaron varias versiones del modelo de detección de objetos en tiempo real YOLO [5], un modelo personalizado y un *ensemble* de modelos. El propósito fue especializar su capacidad, pasando de la detección de objetos cotidianos a la identificación precisa de células redondas. Finalmente, el modelo fue entrenado y evaluado para validar su rendimiento, eficacia y viabilidad.

Este trabajo incluye la creación de una interfaz digital que permite al proyecto trascender la investigación académica para dar continuidad a la misión de DIGIS3 [4]: "impulsar la digitalización de las empresas a través de soluciones tecnológicas avanzadas". La herramienta web desarrollada actúa como un catalizador de la transformación digital para Microptic [2], traduciendo un complejo modelo de inteligencia artificial en una solución práctica que potencia sus capacidades de análisis y establece un precedente para futuras innovaciones.

1.2. Objetivos

Este Trabajo de Fin de Máster tiene como objetivo principal diseñar, desarrollar y validar de un sistema inteligente para la detección automatizada de células redondas en imágenes médicas de muestras de semen humano mediante el empleo de técnicas de visión por computador y aprendizaje profundo. Se de muestra así, la integración de competencias técnicas avanzadas en inteligencia artificial, procesamiento de imágenes y desarrollo de *software*, entre otras habilidades transversales.

Con este proyecto, se abordan los siguientes objetivos específicos:

- Investigar y estudiar casuísticas análogas en la detección de células en conjuntos de imágenes médicas y establecer un estado del arte de la cuestión.
- Realizar un preprocesamiento y anotación de los conjuntos de imágenes médicas.
- Implementar, entrenar y validar diferentes modelos u arquitecturas de modelos, comparar su rendimiento y optimizar hiperparámetros.
- Evaluar cuantitativa y cualitativamente los modelos mediante métricas estándar y la observación visual de los resultados.
- Desarrollar una herramienta web interactiva para visualización, inferencia y exportación de resultados que facilite la validación por parte de expertos biomédicos.
- Garantizar la reproducibilidad y cumplimiento ético y legislativo.
- Documentar exhaustivamente el proyecto.

2. Antecedentes

En este capítulo se expone el problema abordado en este trabajo, sus antecedentes, hipótesis y motivaciones. Asimismo, se revisa la literatura existente para situar el estado del arte y justificar la necesidad de la investigación realizada.

2.1. Contexto y motivación

El análisis de muestras de semen es un pilar fundamental en el campo de la medicina reproductiva. Su relevancia abarca desde el diagnóstico de la infertilidad en parejas y técnicas de reproducción asistida en humanos, hasta la investigación y producción en el sector ganadero. Tradicionalmente, este análisis se ha centrado en la motilidad, concentración y morfología de los espermatozoides.

Sin embargo, un análisis seminal requiere la evaluación de otros componentes celulares, entre los que destacan las células redondas. Estas constituyen un indicador de diagnóstico crucial, ya que abarcan linfocitos, células plasmáticas, macrófagos, mastocitos y células tumorales redondas, cuya presencia puede ser signo de infecciones o inflamaciones del tracto genital, como células germinales inmaduras, que pueden indicar una alteración en el proceso de espermatogénesis [6].

El método estándar para la identificación y conteo de células redondas depende de la agudeza visual de un experto a través de un microscopio; esto lo convierte en un proceso subjetivo, lento, laborioso y cansado para el observador. Estas limitaciones motivan la necesidad de desarrollar alternativas mediante herramientas automatizadas que permitan obtener resultados más rápidos, objetivos y reproducibles, optimizando el espermograma.

El auge de la inteligencia artificial, y en particular las técnicas de visión por computador basadas en aprendizaje profundo, están protagonizando una revolución dentro del mundo de la medicina. La capacidad de las redes neuronales en la detección de patrones complejos ha demostrado ser una herramienta de apoyo en diversas especialidades como son: análisis de imágenes radiológicas o análisis de lesiones cutáneas en dermatología, entre otras. En *computer vision*, modelos de detección de objetos como YOLO (*You Only Look Once*) [5] han demostrado un equilibrio entre

precisión y velocidad de inferencia. Estos algoritmos son capaces de procesar imágenes y localizar múltiples objetos de interés en milisegundos, siendo idóneo en este tipo de trabajos clínicos.

Atendiendo a estas necesidades y casuísticas, por parte de una empresa líder en el análisis espermático, surge la necesidad de integrar estos sistemas inteligentes para la detección de células redondas a la que dio solución el grupo GVIS de la Universidad de León mediante una prueba de concepto [6]. Demostrando así, la capacidad de estas tecnologías en este tipo de tareas complejas.

Aprovechando este hito, surge la motivación de definir una nueva prueba de concepto integrada en un entorno digital, que permita igualar o superar con menos recursos, la anterior prueba de concepto. Esto se aborda a través de diferentes vías:

- Explorar diferentes arquitecturas de YOLO [5], desde modelos oficiales hasta prototipos más recientes que emplean capas de atención.
- Diseñar un modelo personalizado para competir con los modelos existentes.
- Optimización y mejora de la detección: buscar la configuración óptima que maximice diferentes métricas y abordar la detección de artefactos visuales que fueron identificados como una limitación en la PoC previa.
- Desarrollar una herramienta web interactiva que no solo sirva para evaluar imágenes sino que además, dé soporte a expertos.

2.2. Estado del arte e hipótesis

En esta sección se analiza el panorama tecnológico que sirve como fundamento para este proyecto. Se parte de las tecnologías de inteligencia artificial que tienen relevancia en el campo de la medicina como es la visión por computador, se abordan los modelos de detección de objetos utilizados y se revisan las diferentes técnicas de optimización. Finalmente, se realiza un breve estudio sobre la literatura actual en este campo y se define la hipótesis de trabajo.

2.2.1. Visión por computador y modelos de detección de objetos

La inteligencia artificial está en auge y algunos de los campos en los que está tomando protagonismo son el aprendizaje profundo y la visión por computador. Estos son los campos en los que se fundamenta esta prueba de concepto. La visión por computador está revolucionando el análisis de imágenes dadas las capacidades de estos sistemas de interpretar datos visuales en tareas como la clasificación, segmentación semántica o la detección de objetos. La detección de objetos se fundamenta en aprendizaje profundo, mediante el uso de redes neuronales. Los algoritmos pueden identificar patrones complejos en las imágenes con una precisión que no deja de sorprender.

Dentro de este campo, la detección de objetos tiene como objetivo localizar espacialmente (mediante *bounding boxes*) y clasificar cada una de las múltiples instancias de un objeto de interés dentro de una misma imagen. Esta tarea es crucial para este proyecto, pues va a permitir detectar y cuantizar las células redondas en una imagen de esperma humano.

La familia de modelos de YOLO [5] se ha consolidado como un referente para tareas de detección, permitiendo un equilibrio entre precisión y velocidad. Dentro de las 12 arquitecturas de YOLO [5], se emplean las siguientes:

- **YOLOv8:** Adopta un diseño eficiente y sin anclajes (*anchor-free*). Su arquitectura utiliza una versión refinada de módulos basados en *Cross-Stage Partial CSP* y una cabeza desacoplada que separa las tareas de clasificación y regresión, logrando un buen equilibrio entre velocidad y precisión [7] (Figura 2.1).
- **YOLOv9:** Introduce la Información de Gradiente Programable (PGI) y la Red de Agregación de Capas Generalizada y Eficiente (GELAN). Estas innovaciones ayudan a mantener gradientes robustos a través de la red, mejorando la convergencia y la capacidad para detectar objetos pequeños al evitar la pérdida de información [7] (Figura 2.1).
- **YOLOv10:** Elimina la necesidad de la supresión no máxima (NMS), un paso de postprocesamiento que consume tiempo, mediante una novedosa estrategia de asignación dual. Esto reduce significativamente la latencia de inferencia y lo hace más eficiente para la detección en tiempo real [7] (Figura 2.1).

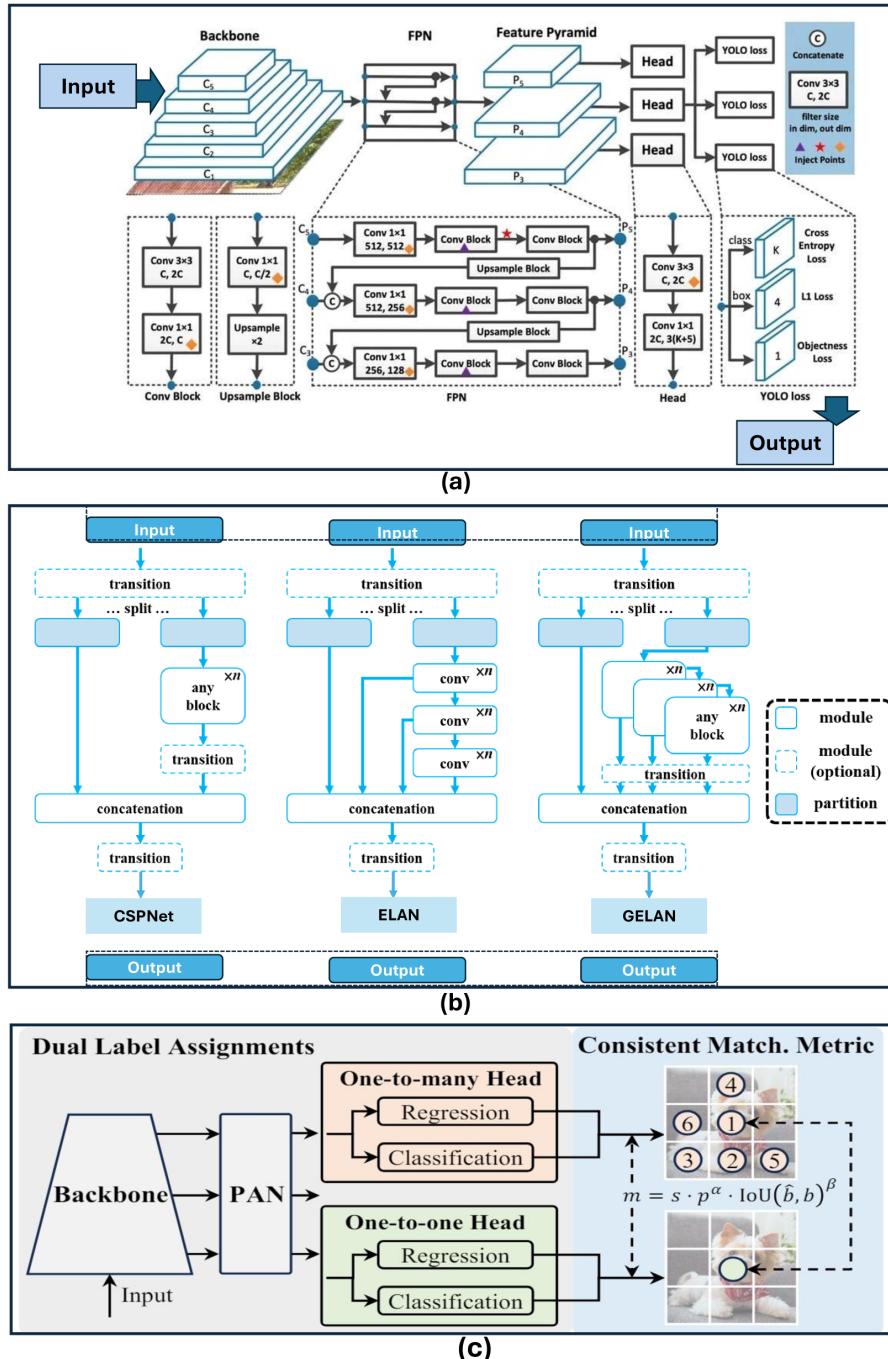


Figura 2.1: Diagramas de arquitectura de YOLOv8, YOLOv9 y YOLOv10

(a) YOLOv8 presenta una estructura principal basada en CSP, un cabezal desacoplado y libre de anclajes (*anchor-free*), y una Red de Pirámide de Características (FPN) optimizada para una extracción eficiente de características a múltiples escalas. (b) YOLOv9 integra la Información de Gradiente Programable con GELAN para una agregación robusta de características. (c) YOLOv10 presenta una estrategia de asignación dual, cabezales ligeros y un submuestreo desacoplado de canal espacial para mejorar la *precision* y velocidad de la inferencia.

- **YOLOv11:** Se enfoca en mejorar la extracción de características mediante el bloque C3k2, el módulo *Spatial Pyramid Pooling-Fast* (SPPF) y la Atención Espacial Paralela (C2PSA). Estas mejoras le permiten un rendimiento más preciso, especialmente en escenarios con objetos ocluidos [7] (Figura 2.2).
- **YOLOv12:** Su principal innovación es el módulo de Atención de Área (A^2), que ajusta dinámicamente el campo receptivo para capturar señales contextuales globales y locales con un costo computacional mínimo. Esto, junto con la red R-ELAN, optimiza la fusión de características y aumenta drásticamente la velocidad de inferencia [7] (Figura 2.2).

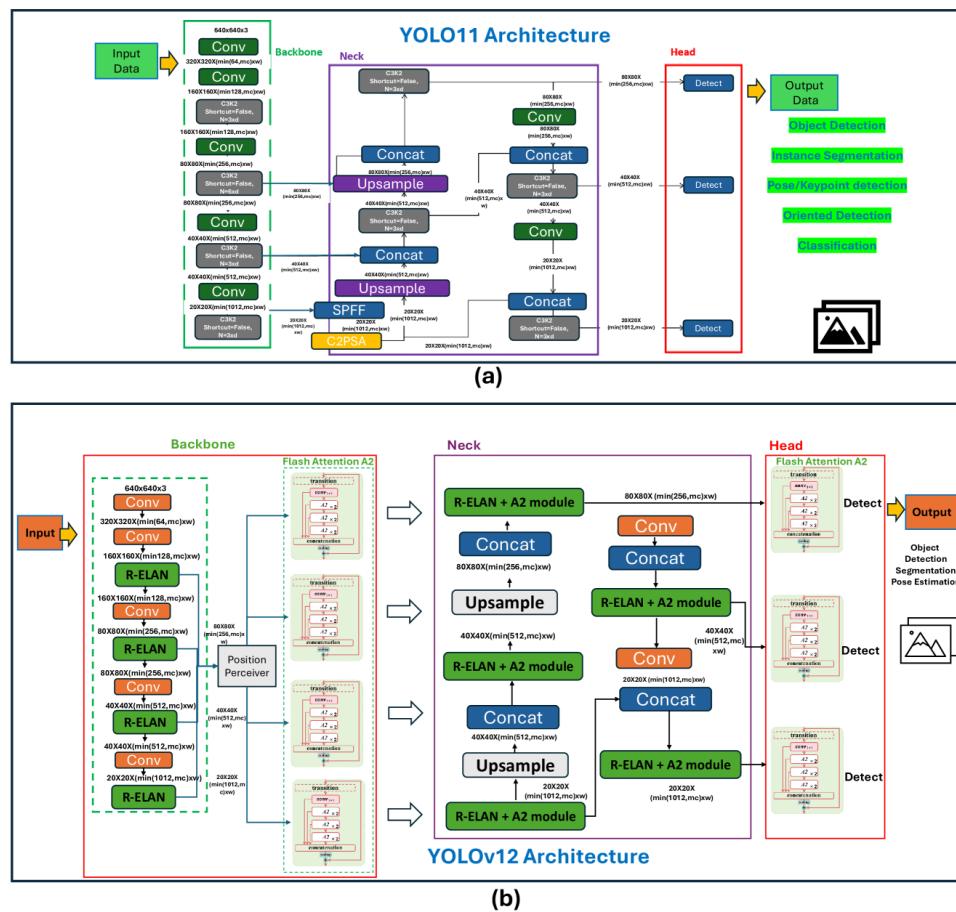


Figura 2.2: Diagramas de arquitectura de YOLOv11 y YOLOv12

(a) YOLOv11 utiliza una arquitectura mejorada que incluye bloques C3k2, SPPF y C2PSA. Estos componentes optimizan la extracción de características en diversas escalas y mejoran la atención espacial, lo que se traduce en una mayor precisión de detección. (b) YOLOv12 avanza sobre este diseño con una arquitectura centrada en la atención. Al integrar módulos de *Area Attention* y bloques R-ELAN, optimiza la combinación de características, logrando un aumento drástico en la velocidad de inferencia para una detección de objetos en tiempo real de vanguardia.

2.2.2. Técnicas de optimización y validación de modelos

Es bien conocida la existencia de diferentes tecnologías para asegurar la robustez y maximizar el rendimiento de los modelos. Algunas de estas tecnologías, son las siguientes:

- **Optimización de hiperparámetros (Optuna):** El *framework* Optuna [8], de optimización automática de hiperparámetros (tales como la tasa de aprendizaje, el *momentum* o regulación), explora de manera inteligente y eficiente el espacio de búsqueda y, al final, devuelve la mejor combinación de hiperparámetros encontrada. Se fundamenta en algoritmos de muestreo bayesiano y mecanismos de poda (*pruning*) que descartan de manera anticipada los ensayos poco prometedores.
- **Aumento de datos (*data augmentation*):** Para mejorar la capacidad de generalización de un modelo y reducir el riesgo de sobreajuste (*overfitting*) en conjuntos de datos limitados se recurre a esta técnica. Esta permite generar nuevas muestras de entrenamiento aplicando transformaciones geométricas (rotaciones, traslaciones, escalado, etc.) y fotométricas (cambios de brillo, contraste, etc.) sobre las imágenes originales, incrementando así la diversidad del conjunto de datos.
- **Validación cruzada (*cross validation*):** Dentro de sus múltiples variantes, la más común consiste en dividir el conjunto de datos en K subconjuntos o pliegues. El modelo se entrena K veces, utilizando en cada iteración un pliegue diferente para la validación y los K-1 restantes para el entrenamiento. El rendimiento final se calcula como la media de los resultados de las K iteraciones, proporcionando una medida de la capacidad de generalización del modelo menos dependiente de una única división de datos. Esta técnica permite conocer si un modelo es fiable y robusto.
- **Ensamblado de modelos (*ensemble*):** Esta técnica avanzada busca mejorar la precisión y la robustez de las predicciones combinando las salidas de múltiples modelos entrenados de forma independiente. La hipótesis subyacente es que los errores de un modelo pueden ser compensados por los aciertos de otros (mediante el promedio o la ponderación de las predicciones individuales), especialmente si los modelos son diversos.

2.2.3. Literatura de la investigación

La inteligencia artificial (IA), y en particular el aprendizaje profundo (*Deep Learning*), están transformando el campo de análisis de imágenes médicas en especialidades como la oncología, la neurología y la oftalmología.

En el ámbito de la andrología, la inteligencia artificial está comenzando a tomar especial relevancia en tareas como la concentración y recuento de espermatozoides, motilidad espermática, morfología espermática, integridad del ADN espermático, entre otras [9]. En el contexto del análisis seminal, la mayor parte de las investigaciones se centran en la automatización del análisis morfológico de los espermatozoides, desarrollando modelos capaces de identificar con alta precisión diferentes partes de la estructura celular (como cabeza, parte intermedia y cola) [10].

Sin embargo, este proyecto presenta un desafío diferente y menos explorado, que a diferencia del análisis morfológico que se centra en un único tipo de célula, la detección de células redondas, como ya se ha visto anteriormente, implica identificar un grupo heterogéneo de células (principalmente leucocitos y células germinales inmaduras) en un entorno complejo y "ruidoso" [1] [11]. La dificultad de identificar estas células redondas, incluso para expertos, y la subjetividad de los métodos manuales tradicionales [12] justifican la necesidad de desarrollar herramientas automatizadas.

Este Trabajo de Fin de Máster se fundamenta en una innovadora prueba de concepto previa, desarrollada por el grupo de investigación GVIS de la Universidad de León en colaboración con la empresa Microptic S.L. [2], en el marco del proyecto europeo DIGIS3 [4]. Dicha prueba de concepto tuvo como objetivo validar la viabilidad de utilizar modelos de *Deep Learning* para la identificación de células redondas en imágenes de semen humano. En ese trabajo inicial, se reentrenaron tres variantes del modelo YOLOv7 utilizando un conjunto de alrededor de 400 imágenes, logrando resultados prometedores en diferentes métricas. Este hito demostró que los detectores de objetos en tiempo real podían ser adaptados con éxito a esta tarea específica a pesar de las dificultades que esto supone.

Este hito sirve de precedente para explorar nuevas y mejoradas arquitecturas y poder superar las métricas obtenidas. Investigaciones de vanguardia han desarrollado modelos de alta precisión para identificar específicamente espermátidas redondas humanas, aunque utilizando muestras purificadas mediante citometría de flujo, lo que simplifica el problema al eliminar el "ruido" de fondo [13].

Por otro lado, trabajos como el modelo ACTIVE abordan un escenario más similar al de este proyecto, al detectar espermatozoides e "impurezas" en videos microscópicos sin procesar [14]. Los resultados para la detección de espermatozoides son realmente buenos, sin embargo, no se puede decir lo mismo para las impurezas (todos los objetos presentes en la muestra que no son espermatozoides). Los propios autores admiten que, en algunos casos, incluso para expertos es difícil distinguir si un objeto es una impureza o un espermatozoide. Esto introduce ambigüedad en los datos de entrenamiento perjudicando el aprendizaje del modelo, especialmente para la clase más difícil de definir ("impurezas") [14].

En síntesis, este proyecto da continuidad al desarrollo tecnológico a partir de la prueba de concepto inicial, además de integrar y buscar superar los desafíos identificados en la literatura científica más reciente. La prueba de concepto sirve como precedente para dar continuidad y actualizar las arquitecturas para el problema de la detección de células redondas en muestras de semen humano, mientras que la literatura reciente pone de manifiesto los desafíos y dificultades en la detección de células heterogéneas.

2.2.4. Hipótesis de trabajo

Considerando el potencial de las arquitecturas YOLO y las diferentes técnicas de optimización, se considera que mediante la exploración de las arquitecturas YOLO que van desde la versión 8 a la 12, junto con el diseño personalizado y la optimización de hiperparámetros, se consiga igualar o mejorar los modelos de la anterior prueba de concepto del grupo GVIS, en las métricas mAP y velocidad de inferencia.

Adicionalmente, se espera que gracias a la arquitectura en desarrollo de YOLOv12 que incorpora mecanismos de atención, el algoritmo no identifique como células redondas a los anillos de Newton. La integración de estos modelos con la herramienta web sirva no solo como método de validación final, sino que también demuestre la viabilidad del sistema para que sea utilizado en un marco clínico real por personal experto.

2.3. Definición del problema

La problemática que da origen a este proyecto no es otra que la ardua y extenuante tarea que le supone a un experto la revisión manual de imágenes médicas de muestras de semen.

Por esta razón, el problema central que aborda este proyecto es diseñar, desarrollar y evaluar un sistema inteligente capaz de automatizar la detección de células redondas sobre un conjunto de imágenes médicas de muestras de semen, utilizando para ello técnicas de visión por computador (*Computer Vision*) y aprendizaje profundo (*Deep Learning*).

Este sistema inteligente debe ser capaz de procesar imágenes microscópicas como las de la Figura 2.3, donde las células objetivo coexisten con una densa población de espermatozoides, diversos artefactos visuales (Anillos de Newton) y restos residuales ajenos al análisis. Asimismo, la dificultad que supone la distinción de células redondas de otros elementos morfológicamente similares (espermatozoides sin cola), esferas de las que se desconoce qué son, etc.

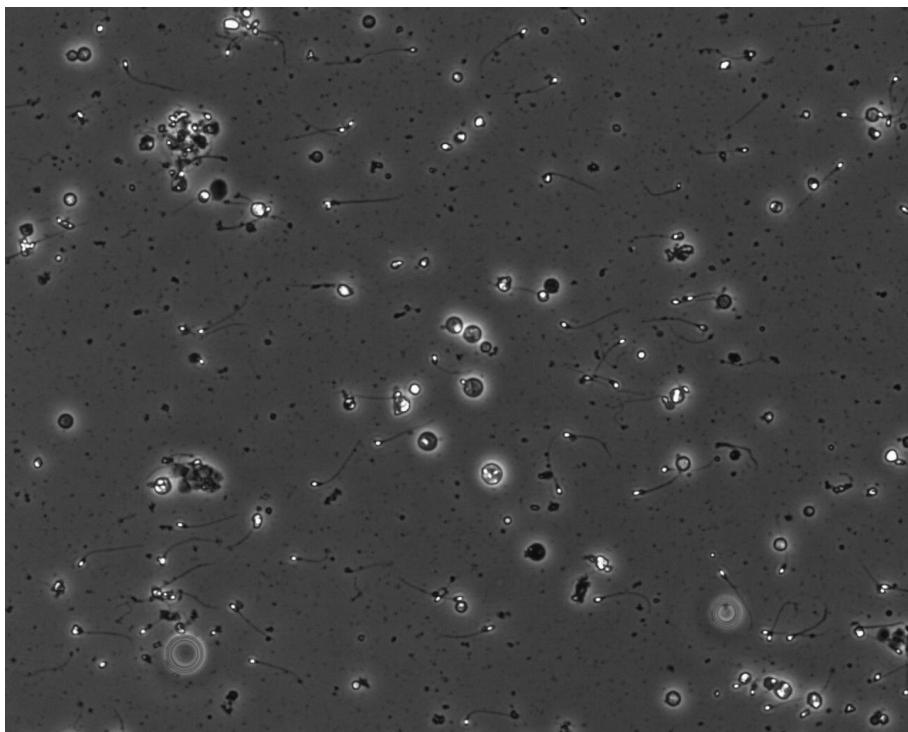


Figura 2.3: Muestra de semen humano sin anotaciones.

La tarea de anotar el dataset presentó una dificultad intrínseca significativa, convirtiéndose en una fuente recurrente de complicaciones durante el desarrollo.

La Figura 2.4 anotada por un experto presenta la problemática de los anillos de Newton. Se verá posteriormente cómo afectan estos objetos a la correcta identificación de células redondas.

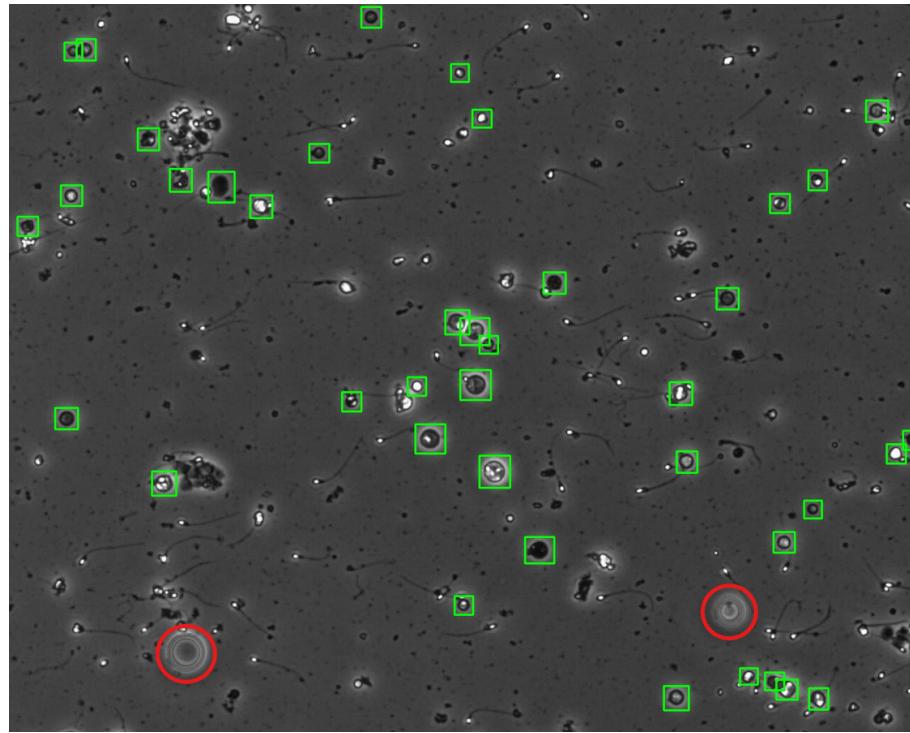


Figura 2.4: Ilustración con anillos de Newton.

Las *bounding boxes* corresponden con las células redondas y los círculos rojos con los anillos de Newton.

Asimismo, la Figura 2.5 ilustra la ambigüedad inherente al proceso de anotación, un desafío corroborado por el feedback de los expertos de Microptic. El análisis de los especialistas revela que, si bien algunas anotaciones son correctas, existe una notable confusión al diferenciar las células redondas de otras estructuras visualmente similares, como espermatozoides con restos de citoplasma o cabezas de espermatozoides sin cola.

De manera crucial, se destacó que la correcta caracterización de ciertas células no es posible mediante la simple inspección visual y requiere de un análisis morfológico o citológico más exhaustivo [1].

Esto evidencia, una vez más, el alto grado de complejidad que supone anotar este tipo de imágenes médicas, incluso dentro del marco de recomendaciones de la OMS [1] [11].

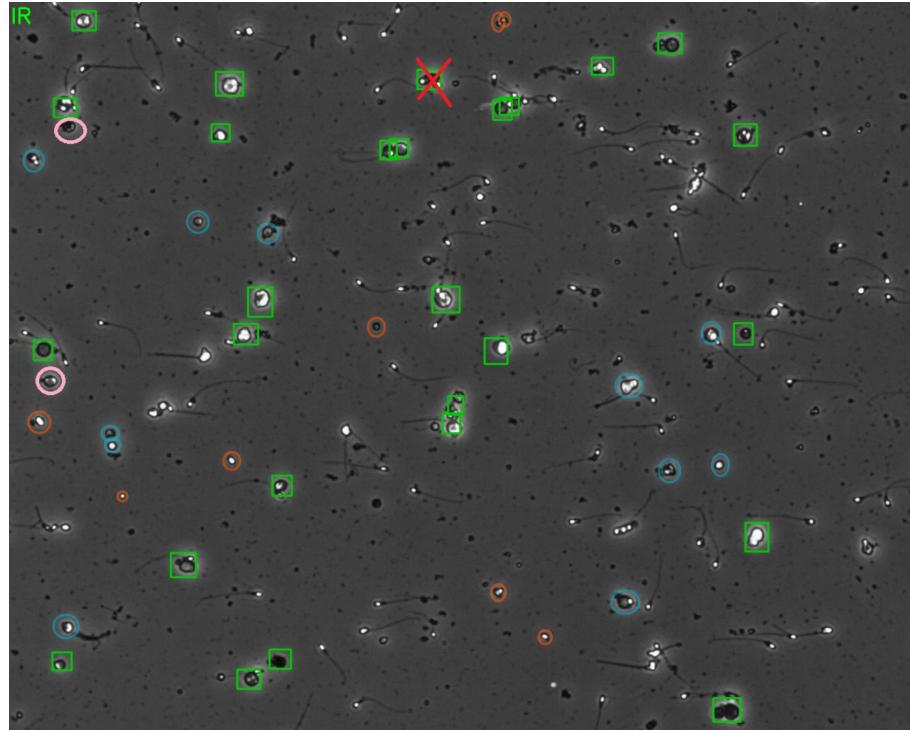


Figura 2.5: Ilustración de anotaciones para *feedback* con experto.

La mayoría de células marcadas con los rectángulos verdes son ciertas células redondas. Los círculos rosas también son células redondas (probablemente germinales, pero es necesario un análisis morfológico específico para saberlo). Los círculos azules marcan correctamente algunas células redondas pero también indican espermatozoides que retienen la mayor parte del citoplasma (esferas grises con un área blanca bien contrastada). Muchos de los círculos naranjas son cabezas de espermatozoides sin cola y las esferas más pequeñas se desconoce que pueden ser.

3. Gestión de proyecto *software*

En este capítulo se definen las tareas a realizar durante el desarrollo, se establece un cronograma y se detalla el presupuesto estimado para la realización del proyecto.

3.1. Alcance del proyecto

3.1.1. Definición del proyecto

El presente Trabajo de Fin de Máster se centra en el diseño, desarrollo y validación de un sistema inteligente para la detección automatizada de células redondas en imágenes médicas de muestras de semen humano. Este proyecto, desarrollado en colaboración con la empresa Microptic S.L., aborda un desafío clave en el campo de la andrología: la necesidad de optimizar y estandarizar el análisis seminal.

Para abordar esta problemática, se desarrolla una solución de *software* basada en aprendizaje profundo, cuyo objetivo es automatizar la detección de células redondas. El alcance se centra en la definición y desarrollo de una prueba de concepto que valida la viabilidad de esta solución tecnológica en un contexto clínico y de investigación.

Para ello, el proyecto abarca desde la gestión y preprocesamiento de los diferentes conjuntos de datos proporcionados por Microptic S.L., hasta el entrenamiento y evaluación de las diferentes arquitecturas de modelos de detección de objetos de YOLO. Como resultado final, se obtiene una herramienta web funcional que permite a los especialistas interactuar con el sistema de inteligencia artificial, obteniendo detecciones de células redondas sobre imágenes médicas de semen humano en tiempo real y, como consecuencia, optimizando el proceso de análisis seminal.

3.1.2. Presupuesto

El presupuesto del proyecto es una estimación que engloba todos los costes asociados a los recursos necesarios para alcanzar los objetivos definidos. Se realiza una valoración económica para simular un entorno profesional y poner valor del trabajo realizado. Los costes se dividen en las siguientes categorías:

Coste de personal

Los costes personales radican en la necesidad de contratar a un ingeniero informático que se encargue del desarrollo técnico del proyecto, así como un *Scrum Master* que supervise el flujo de trabajo y asegure el cumplimiento de los objetivos.

- **Ingeniero Informático:** en España, el sueldo medio de un ingeniero informático es de 30.004,24 € anuales, según el XIX Convenio TIC [15]. En un supuesto de un ingeniero junior con una jornada laboral de 40 horas semanales (1922 horas anuales), se traduce en un coste por hora de 15,61 €/hora.
- **Scrum Master:** en España, el salario base medio es de 43.000,00 € anuales, según los datos de "glassdoor" [16]. En un supuesto de jornada laboral de 40 horas semanales, esto se traduce en un coste por hora de 22,37 €/hora.

En la Tabla 3.1 se detalla el presupuesto estimado para los recursos humanos.

Tabla 3.1: Presupuesto para los recursos humanos

Puesto de trabajo	Horas semanales	Horas totales	Salario (€/hora)	Total
Ingeniero Informático	25	300	15,61	4683,00€
Scrum Master	3	18	22,37	402,66€
Total				5085,66€

Coste del *hardware*

El coste del *hardware* se refiere a los gastos asociados a la adquisición de los equipos necesarios para el desarrollo y ejecución del proyecto. En este caso, se ha optado por emplear los siguientes recursos:

- **Equipo para el Ingeniero Informático:** el coste de un ordenador con las características descritas en la Subsección 5.2.1 es de 765,00€ (según los precios de la web de Lenovo en 2022).
- **Equipo para el Scrum Master:** el coste de un ordenador con las características básicas para ofimática es de 500,00€.
- **Servidor privado:** el arrendamiento de un servidor privado con las características descritas en la Subsección 5.2.1 y mantenimiento incluido, tiene un coste de 0,29€/hora [17].

Tabla 3.2: Desglose de costes de *hardware*

Concepto	Cantidad	Coste
Equipo Ingeniero Informático	1	765,00 €
Equipo <i>Scrum Master</i>	1	500,00 €
Servidor privado ($0,29 \text{ €/h} \times 25 \text{ h}$)	25 h	7,25 €
Total		1272,25€

Coste de *software* y licencias

Para el desarrollo del proyecto se aconseja el empleo de *software* y herramientas específicas que faciliten y optimicen el desarrollo del trabajo. En este caso, se ha optado por emplear GitHub Copilot, una herramienta de Microsoft [18]. El coste de esta herramienta se recoge en la Tabla 3.3.

Tabla 3.3: Presupuesto para los recursos de *hardware*

Concepto	Precio mensual	Meses	Total
GitHub Copilot	10,00€	3	30,00€

Costes indirectos

Los costes indirectos incluyen gastos generales como electricidad, internet e infraestructuras. Se valoran en el 15 % del coste directo del proyecto. El beneficio industrial/empresarial se fija en el 11 %. Sobre la base imponible resultante se aplica el IVA general del 21 % [19].

Coste total

A continuación, se presenta un resumen del presupuesto total estimado para la realización del proyecto. Se desglosa en función de los costes de personal, *hardware*, *software* y licencias, los cuales constituyen los costes directos del proyecto, así como los costes indirectos asociados.

Tabla 3.4: Presupuesto final del proyecto

Tipo de presupuesto	Porcentaje	Coste
Costes personal		5.085,66€
Costes <i>hardware</i>		1.272,25€
Costes software		30,00€
Costes directos		6.387,91€
Costes indirectos	15 %	958,19€
Beneficio industrial	11 %	702,67€
Subtotal		8.048,77€
IVA aplicable	21 %	1.690,24€
Total		9.739,01€

3.2. Plan de trabajo

3.2.1. Metodología

Para la gestión y desarrollo de este proyecto, se ha optado por una metodología ágil basada en Scrum [20]. Este marco de trabajo promueve la entrega incremental de resultados a través de iteraciones cortas y frecuentes, conocidas como sprints, lo que aporta una gran flexibilidad para adaptarse a los desafíos técnicos inherentes al desarrollo de un sistema de visión artificial para la detección de células redondas.

Los sprints para este proyecto tienen una duración de dos semanas, lo que permite un equilibrio adecuado entre la planificación y la ejecución.

Dentro de los roles y responsabilidades, se define al tutor académico como *Scrum Master* y al ingeniero informático como el desarrollador principal.

Para la organización y planificación de las tareas, el proyecto se estructura en torno a un *Product Backlog* (pila de producto). Este *backlog* está compuesto por historias de usuario, que representan cada una de las funcionalidades o tareas a desarrollar. Cada historia de usuario se define con los siguientes atributos:

- Identificador (ID): Un código único para referenciar y realizar el seguimiento de cada tarea.

- Título: Un nombre conciso y descriptivo que resume el objetivo.
- Descripción: Un párrafo que explica con claridad qué trabajo se debe realizar.
- Criterios de aceptación: Una lista de condiciones específicas y verificables que deben cumplirse para que la tarea se considere completada.

3.2.2. Identificación de tareas

Atendiendo a la metodología Scum del proyecto, se definen las siguientes historias de usuario, que se detallan de la Tabla 3.5 a la Tabla 3.10.

ID	1	Título	Planificación, configuración e investigación
Descripción	Sentar las bases del proyecto, configurar el entorno de trabajo y comprender el estado del arte.		
Criterios de aceptación	El entorno de desarrollo (local y remoto) debe estar completamente configurado y probado. Se debe completar una revisión de la literatura sobre modelos de detección de objetos y análisis de imágenes médicas. El plan de proyecto y el análisis de riesgos deben estar documentados.		

Tabla 3.5: Historia de usuario 1 (Planificación, configuración e investigación)

ID	2	Título	Preprocesamiento de datos y modelo base
Descripción	Preparar el conjunto de datos para el entrenamiento y entrenar un primer modelo para establecer una línea de base.		
Criterios de aceptación	El conjunto de datos debe estar completamente anotado y validado. El <i>pipeline</i> de datos, incluyendo el aumento de datos, debe ser funcional. Se debe haber entrenado un modelo YOLO base y obtenido las primeras métricas de rendimiento para validar el proceso.		

Tabla 3.6: Historia de usuario 2 (Preprocesamiento de datos y modelo base)

ID	3	Título	Entrenamiento y optimización
Descripción	Entrenar y comparar diferentes arquitecturas de modelos, optimizando sus hiperparámetros para obtener el mejor rendimiento.		
Criterios de aceptación	Se deben haber entrenado y evaluado al menos dos arquitecturas de modelos diferentes. La optimización de hiperparámetros debe estar completada y documentada. Se debe realizar una validación cruzada para asegurar la generalización y evitar el sobreajuste.		

Tabla 3.7: Historia de usuario 3 (Entrenamiento y optimización)

ID	4	Título	Modelos avanzados y evaluación comparativa
Descripción	Desarrollar y entrenar un modelo <i>ensemble</i> y un modelo personalizado. Evaluar cuantitativamente todos los modelos (base, <i>ensemble</i> y personalizado) para seleccionar el mejor.		
Criterios de aceptación	El modelo <i>ensemble</i> y el modelo personalizado deben estar implementados y entrenados. Se debe realizar una evaluación comparativa con métricas (mAP, precisión, <i>recall</i>) y documentar los resultados para justificar la selección del modelo final.		

Tabla 3.8: Historia de usuario 4 (Modelos y evaluación comparativa)

ID	5	Título	Desarrollo de la herramienta web interactiva
Descripción	Diseñar, implementar y probar la herramienta web con <i>Streamlit</i> , integrando los diferentes modelos de YOLO para permitir la detección de células en tiempo real.		
Criterios de aceptación	La herramienta web debe ser completamente funcional: permitir la carga de imágenes (individuales y por lotes), seleccionar el modelo, visualizar las detecciones con <i>bounding boxes</i> , mostrar métricas y permitir la descarga de resultados. La aplicación debe ser estable y estar documentada en el manual de usuario.		

Tabla 3.9: Historia de usuario 5 (Desarrollo de la herramienta web interactiva)

ID	6	Título	Documentación final y entrega
Descripción	Finalizar la redacción de la memoria del TFM y preparar todos los materiales para la entrega y defensa del proyecto.		
Criterios de aceptación	La memoria del TFM debe estar completamente redactada, incluyendo todos los capítulos, anexos y bibliografía. El código del proyecto debe estar documentado y subido a un repositorio. La presentación para la defensa final debe estar preparada.		

Tabla 3.10: Historia de usuario 6 (Documentación final y entrega)

3.2.3. Estimación de tareas

Esta sección presenta la planificación del proyecto (Tabla 3.11), detallando las tareas a realizar y el cronograma previsto para su ejecución.

Tabla 3.11: Estimación de tareas del proyecto

ID	Historia de usuario	Tareas principales	Semanas
1	Planificación, configuración e investigación	<ul style="list-style-type: none"> - Investigación del estado del arte - Configuración de entornos (local/remoto) - Definición detallada del plan de proyecto y riesgos - Análisis inicial del <i>dataset</i> 	24-25
2	Preprocesamiento de datos y modelo base	<ul style="list-style-type: none"> - Anotación, validación y aumento de datos - Desarrollo del <i>pipeline</i> de datos - Entrenamiento y evaluación de un modelo inicial 	26-27
3	Entrenamiento y optimización	<ul style="list-style-type: none"> - Entrenamiento de múltiples arquitecturas YOLO - Optimización de hiperparámetros - Validación cruzada y análisis de resultados 	28-29
4	Modelos avanzados y evaluación	<ul style="list-style-type: none"> - Desarrollo de modelo <i>ensemble</i> - Implementación de modelo personalizado - Evaluación comparativa 	30-31
5	Desarrollo de la herramienta web	<ul style="list-style-type: none"> - Implementación del <i>backend</i> (lógica de inferencia) - Creación del <i>frontend</i> (interfaz de usuario) - Integración y pruebas funcionales 	32-33
6	Documentación final y entrega	<ul style="list-style-type: none"> - Redacción de la memoria del TFM - Documentación del código y repositorio 	34-35

3.2.4. Planificación de tareas

Atendiendo a la metodología agile Scrum, el desarrollo del proyecto se ha estructurado en seis *sprints* de dos semanas cada uno. Cada *sprint* se corresponde con una historia de usuario, permitiendo un desarrollo lógico y secuencial. En el anexo Apéndice B se detalla mediante del diagrama de Gantt [21] la planificación de cada *sprint*, con una fecha final objetivo que corresponde con la entrega del día 5 de septiembre de 2025.

A continuación, se resume el enfoque de cada sprint:

- Investigación y Configuración (Semanas 24-25): Esta fase inicial se centra en establecer las bases teóricas y técnicas del proyecto. Incluye una revisión del estado del arte, la configuración de los entornos de desarrollo (local y remoto) y un análisis inicial del *dataset* para comprender sus características.
- Preprocesamiento y Modelo Base (Semanas 26-27): Se preparan los datos para el entrenamiento. Se revisan y gestionan los diferentes conjuntos de datos. Finaliza con el entrenamiento de un primer modelo que sirve como referencia.
- Entrenamiento y Optimización (Semanas 28-29): Se entrena y comparan las diferentes arquitecturas de YOLO. Se aplica la optimización de hiperparámetros y se utiliza la validación cruzada para asegurar que los modelos generalicen correctamente y no sufran de sobreajuste.
- Modelos Avanzados y Evaluación (Semanas 30-31): Se desarrolla un modelo *ensemble* que combina las fortalezas de varios modelos, y se realiza la evaluación comparativa final.
- Desarrollo de la Herramienta Web (Semanas 32-33): Se dedica por completo a la creación de la aplicación interactiva con *Streamlit*. Se implementa tanto el backend (lógica de inferencia) como el frontend (interfaz de usuario).
- Documentación y Entrega Final (Semanas 34-35): La última fase se centra en consolidar todo el trabajo. Incluye la redacción final de la memoria del TFM y la documentación del código para garantizar su reproducibilidad.

3.3. Gestión de recursos

La gestión de recursos implica la especificación, asignación y supervisión de los recursos necesarios para llevar a cabo el proyecto de manera eficiente.

3.3.1. Especificación y asignación de recursos

Los recursos disponibles para el desarrollo de este proyecto se clasifican en tres grupos:

- **Recursos económicos:** El presupuesto económico viene detallado en la Subsección 3.1.2, y asciende a un total de 9.739,01€.
- **Recursos físicos:**
 - Equipo para Scrum Master
 - Equipo para el desarrollador
 - Servidor remoto con GPU Tesla T4 proporcionado por GVIS
 - Instalaciones: luz, internet, material de oficina, etc
- **Recursos humanos:**
 - *Scrum Master*: Tutor académico
 - Desarrollador: Ingeniero Informático
 - Experto de Microptic S.L.
 - Soporte de infraestructura (GVIS): Provisión y operación del servidor remoto.

La asignación de recursos se realiza de la siguiente manera: cada rol humano dispone del equipo informático necesario para sus funciones. El desarrollador utiliza su equipo local para el preprocesado de datos, desarrollo del *pipeline* de metodológico y el desarrollo de la aplicación web con *Streamlit*. El servidor remoto con GPU Tesla T4 se reserva para las tareas de mayor coste computacional, como el entrenamiento del modelo pesado (YOLOv12l). Se solicita puntualmente una consulta con el experto de Microptic S.L. para resolver dudas técnicas en las anotaciones de las imágenes. Finalmente, las instalaciones garantizan el soporte operativo (luz, internet) para todas las actividades.

3.4. Gestión de riesgos

Se identifican, analizan y gestionan los riesgos potenciales que podrían afectar al desarrollo exitoso del proyecto.

3.4.1. Identificación y análisis de riesgos

Para una mayor claridad e interpretación, se elabora una tabla que detalla los riesgos identificados, su probabilidad de ocurrencia (prob.), el impacto potencial, las causas subyacentes y las estrategias de gestión propuestas.

Tabla 3.12: Análisis de riesgos del proyecto

ID	Riesgo	Categoría	Probabilidad	Impacto	Causas	Gestión
1	Etiquetado subjetivo de células	Datos	Alta	Alto	Subjetividad en la identificación de células, generando inconsistencias en el etiquetado	Implementar validación y re-etiquetado por expertos para consolidar un <i>ground truth</i> robusto
2	Artefactos confundidos con células	Datos	Baja	Bajo	Presencia de artefactos visuales (ej. anillos de Newton) que se confunden con células redondas	Elevar umbral de confianza de los modelos y evaluarlos en imágenes con artefactos para seleccionar las mejores arquitecturas
3	Dataset limitado	Datos	Media	Alto	La calidad, cantidad o representatividad del dataset puede ser insuficiente o contener sesgos	Aplicar técnicas de <i>data augmentation</i> para incrementar la diversidad y mejorar la generalización
4	Definición ambigua de objetivos	Planificación	Media	Alto	Objetivos mal definidos o cambiantes que dificultan la planificación	Revisión detallada y continua de los objetivos
5	Cambios no planificados en requisitos	Planificación	Baja	Medio	Cambios inesperados	Gestión de cambios y documentación detallada
6	Desviación en los plazos	Planificación	Alta	Medio	Estimaciones optimistas y retrasos en tareas críticas	Monitorización continua
7	Errores de código o integración	Implementación	Alta	Bajo	Escasez de pruebas, errores humanos	Realización continua de pruebas y control de versiones Apéndice A
8	Faltan recursos técnicos o humanos	Implementación	Alta	Bajo	Limitaciones de <i>hardware</i> en el uso de GPU	Se priorizan modelos ligeros, se ajusta el <i>batch size</i> a la VRAM disponible; se externaliza a un servidor privado
9	Riesgo de <i>overfitting</i>	Implementación	Media	Medio	Los modelos sobreajusten (<i>overfitting</i>) los datos de entrenamiento y no generalicen correctamente a nuevos conjuntos de imágenes	Empleo de pruebas de validación, técnicas de regularización y evaluación final sobre múltiples conjuntos de test para verificar la generalización
10	Retrasos por complejidad técnica no prevista	Implementación	Media	Bajo	Subestimación de la complejidad, falta de experiencia	Reuniones breves de alineación, formación continua para el equipo
11	Presión presupuestaria	Externo	Baja	Alto	Costes de <i>hardware</i> y <i>software</i> puntuales	Definir un tope de gasto y priorizar ejecuciones críticas
12	Eventos imprevistos	Externo	Baja	Alto	Pandemia, apagón eléctrico o de internet, etc	Plan de contingencia local-remoto y seguro adecuado
13	Dependencia de terceros	Externo	Media	Medio	Alquiler de GPU no disponible o retraso en entregas de <i>hardware</i> , servidores caídos, etc	Diversificación de proveedores y contratos sólidos con cláusulas que penalizan el incumplimiento

3.5. Legislación y normativa

En el marco de ejecución de este proyecto, se ha llevado a cabo un riguroso cumplimiento de la legislación y normativa vigente. A continuación, se detalla cómo el proyecto se ajusta y adhiere a las leyes pertinentes:

- **Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales.** [22]

Este proyecto respeta plenamente la Ley Orgánica 3/2018, la cual reconoce el derecho fundamental a la protección de datos personales. La creación y tratamiento del *dataset* de imágenes médicas se ha realizado conforme a las disposiciones de la ley, asegurando la legalidad en el tratamiento de datos biomédicos. Se han obtenido los permisos explícitos necesarios para el uso de las imágenes, garantizando la privacidad y anonimización de los datos de los pacientes.

- **Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016, relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga la Directiva 95/46/CE (Reglamento general de protección de datos).** [23]

La creación y tratamiento del *dataset* de imágenes médicas en este proyecto se ha realizado conforme a los principios del RGPD, garantizando la legalidad y transparencia en el tratamiento de datos biomédicos. Todas las imágenes han sido previamente anonimizadas y se han implementado medidas técnicas y organizativas para asegurar la seguridad y privacidad de los datos, cumpliendo así con las exigencias del RGPD para datos de salud considerados de categoría especial.

- **Reglamento (UE) 2024/1689 del Parlamento Europeo y del Consejo, de 13 de junio de 2024, por el que se establecen normas armonizadas en materia de inteligencia artificial.** [24]

Este reglamento establece normas armonizadas para garantizar la seguridad, ética y transparencia en el desarrollo y aplicación de sistemas de IA en la Unión Europea. En el contexto de este TFM, el sistema desarrollado se considera una herramienta de investigación y apoyo a la evaluación biomédica. No está concebido ni validado para toma de decisiones clínicas autónomas, por lo

que su uso actual no se presenta como IA de alto riesgo. No obstante, para alinearse con los requisitos se incorporan las siguientes medidas: documentación completa, evaluación de riesgos y validación, trazabilidad y registro para una posterior reproducibilidad.

- **Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia. [25]**

En conformidad con el Real Decreto Legislativo 1/1996, el proyecto respeta la normativa sobre propiedad intelectual. Se ha optado por utilizar únicamente código y herramientas de *software* libre y de código abierto para garantizar el cumplimiento de la normativa en materia de propiedad intelectual.

4. Metodología

En esta sección se describen de forma reproducible, las tres etapas clave experimentales implementadas de forma secuencial:

- **Fase 1:** Gestión y preprocesado del *dataset*.
- **Fase 2:** Diseño experimental, entrenamiento y evaluación de modelos.
- **Fase 3:** Despliegue de la herramienta web.

4.1. Gestión y preprocesado del *dataset*

La base de cualquier sistema de inteligencia artificial reside en la calidad de los datos. Esta primera fase se centra en el estudio y procesamiento del conjunto de datos (Sección 5.1), que servirá como fuente de verdad (*ground truth*) en las etapas posteriores.

- **Adquisición y consolidación:** El punto de partida es el conjunto de imágenes médicas proporcionado por Microptic S.L. [2] Se realiza un análisis para comprender su estructura, el formato original (PASCAL VOC) y la calidad de las anotaciones. Un conjunto de datos para entrenamiento y tres para *test*.
- **Validación por expertos:** Atendiendo a la revisión de *feedback* proporcionada por expertos sobre los conjuntos de *test*, se procede con el reetiquetado (mediante LabelImage [26]) para asegurar una mejor precisión de las anotaciones. Como no se dispone de *feedback* sobre el conjunto de entrenamiento y, además, se quiere comparar con la prueba de concepto inicial, este no se modifica ni se pide una nueva valoración de expertos sobre el conjunto de datos de entrenamiento.
- **Estructuración del dataset:** Para entrenar las diferentes arquitecturas de YOLO, el conjunto de datos de entrenamiento original, se divide el conjunto de datos en entrenamiento (80 %) y validación (20 %). Para ello, se emplean una semilla determinada para garantizar que los experimentos sean reproducibles.
- **Nuevo formato de datos:** Una vez se ha organizado la estructura de carpetas, las anotaciones en formato PASCAL VOC se convierten a formato YOLO,

que es requerido para la metodología actual. Adicionalmente, se generó una versión en formato COCO para garantizar la interoperabilidad y facilitar futuras aplicaciones con otros marcos de trabajo.

4.2. Diseño experimental, entrenamiento y evaluación de modelos.

Esta fase constituye el núcleo de la investigación, donde se exploran diferentes arquitecturas y configuraciones para obtener el modelo óptimo. El detalle completo previo y durante la etapa de entrenamiento de los diferentes modelos se recoge en la Sección 5.3.

- **Arquitectura:** El código de esta fase se estructura de forma altamente modular, facilitando el mantenimiento y escalabilidad. La lógica completa se encapsula en clases y librerías reutilizables, manteniendo los cuadernos de *Jupyter* limpios y con un hilo argumental claro [27].
- **Optimización de modelos:** Se realiza una búsqueda sistemática mediante el *framework* Optuna [28] para la optimización automática de hiperparámetros.
- **Entrenamiento y validación:** Para obtener modelos que generalicen lo mejor posible, se emplean las técnicas:
 - **Aprendizaje por Transferencia (*Fine-tuning*):** Se emplean los pesos preentrenados de las diferentes arquitecturas de YOLO. Sobre esta base de conocimiento, los modelos son reentrenados específicamente con el dataset de células redondas. Este enfoque acelera el entrenamiento y permite alcanzar un rendimiento elevado.
 - **Aumento de datos (*data augmentation*):** Durante el proceso de entrenamiento, se aplican transformaciones para enriquecer la diversidad de los datos y reducir el *overfitting*.
 - **Validación cruzada (*k-fold cross-validation*):** Se emplea para evaluar de forma robusta la capacidad de generalización de los modelos y la configuración de los hiperparámetros.
- **Exploración de enfoques avanzados:** Adicionalmente, se aborda el problema desde otras perspectivas:

- **Ensamblado de modelos:** Se combinan las predicciones de diferentes modelos individuales para crear un nuevo sistema de detección de células redondas.
- **Modelo personalizado:** Se diseña una arquitectura de red neuronal propia.
- **Evaluación:** Se realiza una evaluación cuantitativa y cualitativa de los resultados (Ver Capítulo 6).
 - **Evaluación cuantitativa:** Se estudia el rendimiento individual de los modelos sobre los diferentes conjuntos de *test*. Con este fin, se emplean las métricas: *precision*, *recall*, mAP@0.5, mAP@0.5:0.95, el tiempo de inferencia y el F1-confianza (Sección 5.4).
 - **Evaluación cualitativa:** Se evalúan los modelos en un entorno problemático, como son las imágenes con anillos de Newton (Sección 2.3). Asimismo, se realiza una comparativa de resultados respecto a algunas imágenes de resultados obtenidos en la prueba de concepto inicial.

4.3. Desarrollo y despliegue de la herramienta web

La tercera y última fase del proyecto es dar una solución tangible y accesible para los usuarios finales (expertos) (Capítulo 7).

- **Desarrollo de la interfaz:** Se define una aplicación interactiva empleando el *framework* de *Streamlit*, con el fin de que los expertos médicos puedan interactuar de forma intuitiva con el sistema de detección de células redondas.
- **Funcionalidades:** La prueba de concepto se integra con una serie de funcionalidades como la carga de imágenes y anotaciones asociadas (opcional), selección de modelo de inferencia e intervalo de confianza de las predicciones, visualización de las detecciones, comparativa de resultados con el *ground truth*, exportación de predicciones y métricas sobre un conjunto de imágenes.
- **Arquitectura de software:** La aplicación se diseña con un enfoque modular, que separa la lógica de procesamiento de la interfaz de usuario, garantizando la escalabilidad y la facilidad de mantenimiento [27].

5. Experimentación

5.1. *dataset*

El *dataset* inicial proporcionado por la empresa Microptic [2] está constituido por un conjunto de *train* y *test* en formato PASCAL VOC.

Tabla 5.1: Distribución del *dataset* original

Partición	Resoluciones (px)	Nº imágenes (resolución)	Nº imágenes	Sin instancias	Escala grises
Train	1280 × 1024	356	373	23	3 canales
	768 × 616	17			
Test	1280 × 1024	87	94	6	3 canales
	768 × 616	7			

El conjunto de datos inicial que nos proporciona Microptic [2] está compuesto por 373 imágenes para *train* y 94 para *test*, de las cuales no presentan anotaciones: 23 imágenes de *train* y 6 de *test*. El conjunto de entrenamiento se divide utilizando la función `train_test_split` de *scikit-learn*, reservando el 80% (298 imágenes) para entrenamiento y el 20% (75 imágenes) para validación. Esta división se realiza de forma aleatoria pero reproducible, fijando la semilla (`random_state = 42`) para garantizar la consistencia de los resultados.

Adicionalmente, la empresa proporciona un conjunto de datos del *test* reevaluado por expertos del dominio. Este conjunto se reetiqueta utilizando la herramienta *LabelImg* [26] en formato PASCAL VOC. Las correcciones afectan a un total de 60 imágenes, incrementando el número de instancias de 1273 a 1412.

Asimismo, se incluyeron dos conjuntos adicionales denominados *test2* y *test3*, inicialmente sin anotaciones pero que contenían *bounding boxes* generados por modelos YOLO [5] preentrenados por el grupo GVIS de la Universidad de León. Estas predicciones fueron posteriormente corregidas y validadas por expertos, proporcionando dos conjuntos adicionales para evaluación. La composición final del *dataset* se presenta en la Tabla 5.2.

Tabla 5.2: Resumen del *dataset* de actuación

Partición	Imágenes	Instancias	1280×1024	768×616	Escala grises
<i>Train</i>	298	3934	282	16	3 canales
<i>Validation</i>	75	878	74	1	3 canales
<i>Original_test</i>	94	1273	87	7	3 canales
<i>Test</i>	94	1412	87	7	3 canales
<i>Test2</i>	10	144	0	10	1 canal
<i>Test3</i>	59	1135	56	3	1 canal

El conjunto de *train*, *validation* y *test* están constituidos por imágenes RGB de 3 canales, mientras que las imágenes de los conjuntos de *test2* y *test3* están compuestos por imágenes en escala de grises de un único canal.

Para el entrenamiento de diferentes arquitecturas de detección de objetos, se convierten los formatos de PASCAL VOC a YOLO y YOLO a COCO. Para más detalle, la información relativa al preprocesamiento del *dataset* se encuentra en el documento `preprocesamiento.ipynb` del repositorio [27].

5.2. Entorno de desarrollo

Se utilizan dos entornos de desarrollo complementarios, garantizando la reproducibilidad y escalabilidad de los resultados obtenidos.

5.2.1. *Hardware*

hardware Local

El entorno principal de desarrollo es un equipo *Lenovo IdeaPad Gaming 3* con las siguientes especificaciones técnicas:

- **Procesador:** AMD Ryzen 5 5600H with Radeon Graphics
 - Velocidad base: 3,30 GHz
 - Núcleos físicos: 6
 - Procesadores lógicos: 12

- Caché L1: 384 kB
- Caché L2: 3,0 MB
- Caché L3: 16,0 MB
- Virtualización: Habilitada
- **GPU:** NVIDIA GeForce RTX 3050 Laptop GPU
 - Memoria dedicada: 4,0 GB GDDR6
 - Arquitectura: Ampere
 - Soporte CUDA: 12,9
 - *Driver version:* 576.02
- **Memoria RAM:** 16 GB DDR4 SODIMM
 - Velocidad: 3200 MHz
 - Configuración: 2 módulos de 8 GB
- **Almacenamiento:** SSD NVMe PCIe Gen3 x4
 - Capacidad: 512 GB
 - Modelo: Micron MTFDHBA512QFD

Servidor privado

Como entorno complementario se ha empleado un servidor remoto proporcionado por el grupo GVIS de la Universidad de León.

- **GPU:** Tesla T4 con 15 GB de memoria
- **RAM del sistema:** 12,7 GB
- **Almacenamiento temporal:** 78,2 GB SSD

5.2.2. *software y Frameworks*

El desarrollo se realizó empleando el siguiente *stack* tecnológico:

- **Sistema Operativo:** Windows 11
- **Entorno Python:** Miniconda3 (entorno TFM)

- **IDE:** Microsoft Visual Studio Code con Github Copilot
- **CUDA Toolkit:** Versión 12.9
- **Frameworks principales:**
 - PyTorch 2.6.0 con soporte CUDA 12.6
 - Ultralytics 8.3.177
 - OpenCV para procesamiento de imágenes
 - Optuna para optimización de hiperparámetros
 - Streamlit para desarrollo de interfaces web
 - Pandas para análisis y manipulación de datos
 - Matplotlib para visualización de datos
 - Scikit-learn para métricas y otras utilidades de *Machine Learning*
 - Jupyter Notebook para desarrollo y análisis interactivo
 - Linter ruff para mantener un código limpio, coherente y más fácil de mantener.

Para el despliegue del proyecto se recomienda tener todas las dependencias del `requirements.txt` [27], el cual recoge todas las librerías y versiones necesarias para la correcta ejecución del entorno.

5.3. Configuraciones

En esta sección se abordan los aspectos más relevantes del entrenamiento, validación y desarrollo de metodologías avanzadas para la detección de células redondas.

5.3.1. Entrenamiento

Para abordar el proceso de entrenamiento de forma óptima, previamente se emplea el *framework* Optuna [28] con las configuraciones de parámetros adjuntas en la Tabla 5.3 y maximizando la métrica mAP@0.5:0.95. Cabe destacar que durante todo el procedimiento se garantiza la reproducibilidad y cumplimiento con las normas éticas y legales.

Toda la información relativa a esta etapa se encuentra en el documento de configuraciones (config.py) del repositorio del proyecto [27]. Recordar que al tener como objeto la detección de células redondas, el número de clases es uno.

Tabla 5.3: Parámetros principales de entrenamiento para cada modelo

Model	Nº Trials	Epoch Optuna	Epoch Train	Batch Size	Image Size
YOLOv8s	10	25	40	12	704
YOLOv9s	7	25	40	10	704
YOLOv10s	7	25	40	10	704
YOLOv11s	7	25	40	10	704
YOLOv12s	9	25	40	7	704
YOLOv12l	6	25	40	7	704
<i>custom</i>	10	25	40	12	704

Como resultado de aplicar Optuna para cada arquitectura, se obtienen las mejores configuraciones de hiperparámetros que se recogen en Tabla 5.4.

Tabla 5.4: Principales hiperparámetros óptimos seleccionados para cada modelo

Modelo	lr0	lrf	momentum	weight_decay	optimizer
YOLOv8s	0,00540	0,00399	0,90621	0,00010	SGD
YOLOv9s	0,00023	0,00153	0,84564	0,00033	AdamW
YOLOv10s	0,00540	0,00399	0,90621	0,00010	SGD
YOLOv11s	0,00023	0,00153	0,84564	0,00033	AdamW
YOLOv12s	0,00023	0,00932	0,91627	0,00087	AdamW
YOLOv12l	0,00019	0,00196	0,85495	0,00029	SGD
<i>custom</i>	0,00153	0,00111	0,89113	0,00015	AdamW

Para mejorar la capacidad de generalización de los modelos y prevenir el sobreajuste, se aplicaron técnicas de aumento de datos (*data augmentation*) durante el entrenamiento. Los hiperparámetros utilizados se muestran en la Tabla 5.5. Los parámetros de calentamiento (*warmup_epochs* y *warmup_momentum*) controlan la fase inicial del entrenamiento, donde el optimizador incrementa gradualmente

la tasa de aprendizaje durante 5 épocas, partiendo de un *momentum* de 0.75. Los parámetros de transformación geométrica incluyen: *degrees*, que permite rotaciones aleatorias de hasta 45 grados; *translate*, que permite desplazamientos de hasta el 10 % de las dimensiones de la imagen; y *scale*, que permite cambios de escala de hasta $\pm 6\%$. Las transformaciones de volteo horizontal y vertical (*fliplr* y *flipud*) se aplican con una probabilidad del 50 % cada una. Finalmente, el parámetro *mosaic* (técnica que combina cuatro imágenes en una) se desactivó (valor 0) para evitar distorsionar el contexto espacial de las células redondas, que es crucial para su correcta identificación en imágenes microscópicas.

Tabla 5.5: Hiperparámetros de *data augmentation*

Parámetro	Valor	Parámetro	Valor
warmup_epochs	5	flipud	0,5
warmup_momentum	0,75	fliplr	0,5
degrees	45	mosaic	0
translate	0,1	close_mosaic	0
scale	0,06		

Todos los modelos han sido entrenados y evaluados con el limitado *hardware* personal (Subsección 5.2.1), exceptuando el modelo YOLov12l que demanda el uso de un servidor privado proporcionado por la Universidad de León (Subsección 5.2.1).

A continuación, se exponen e interpretan los resultados obtenidos durante los entrenamientos:

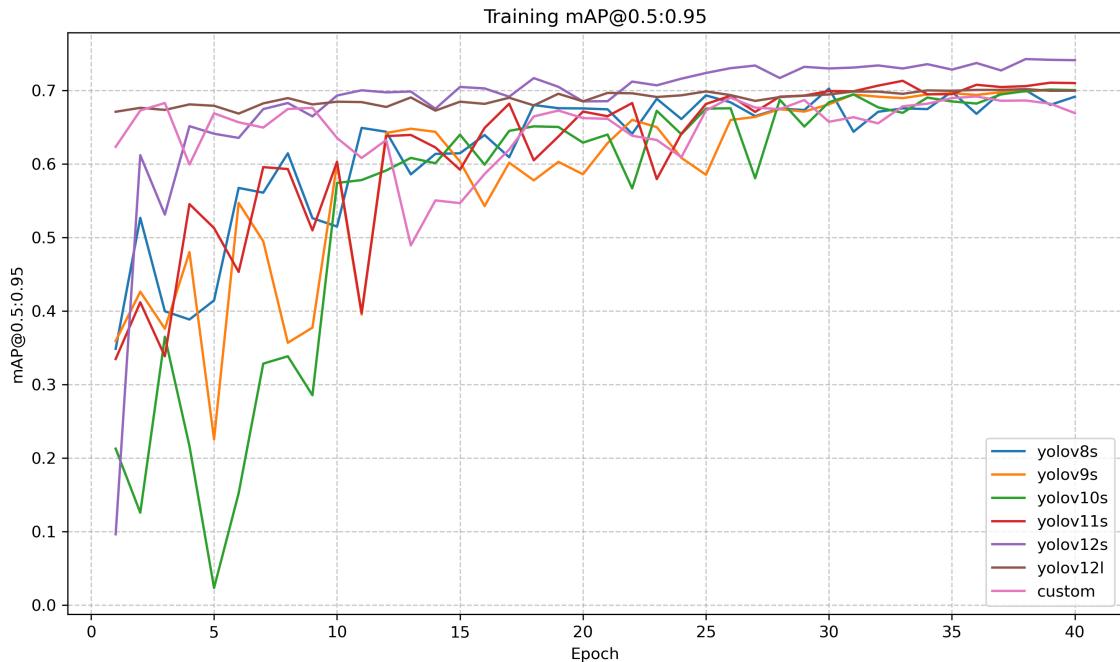


Figura 5.1: mAP@0.5:0.95 durante el proceso de entrenamiento.

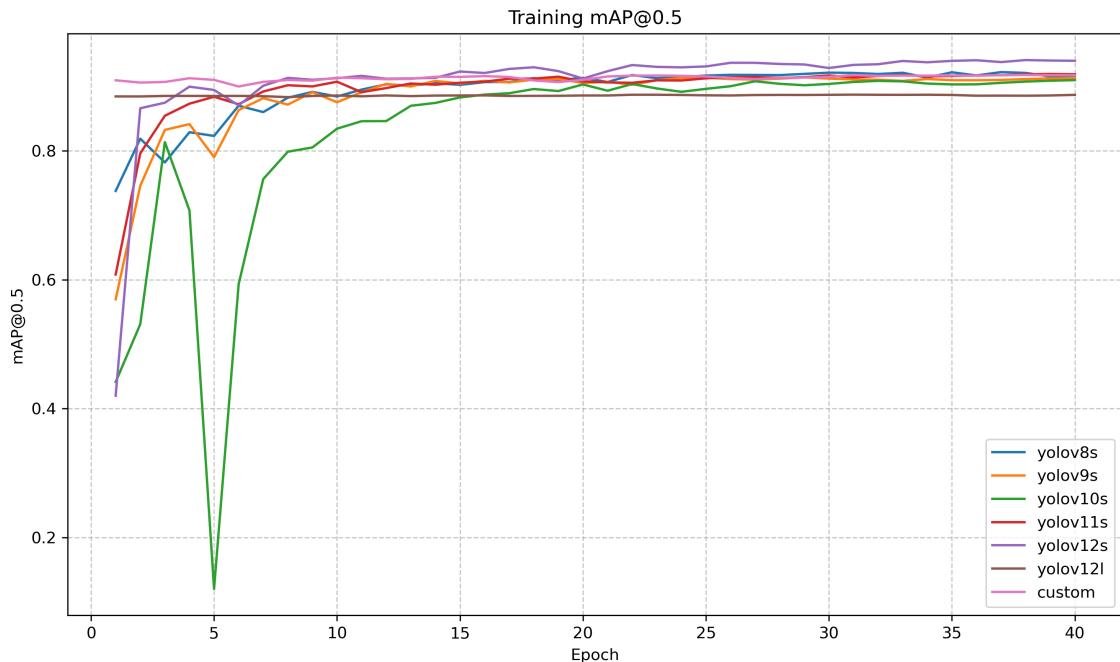


Figura 5.2: mAP@0.5 durante el proceso de entrenamiento.

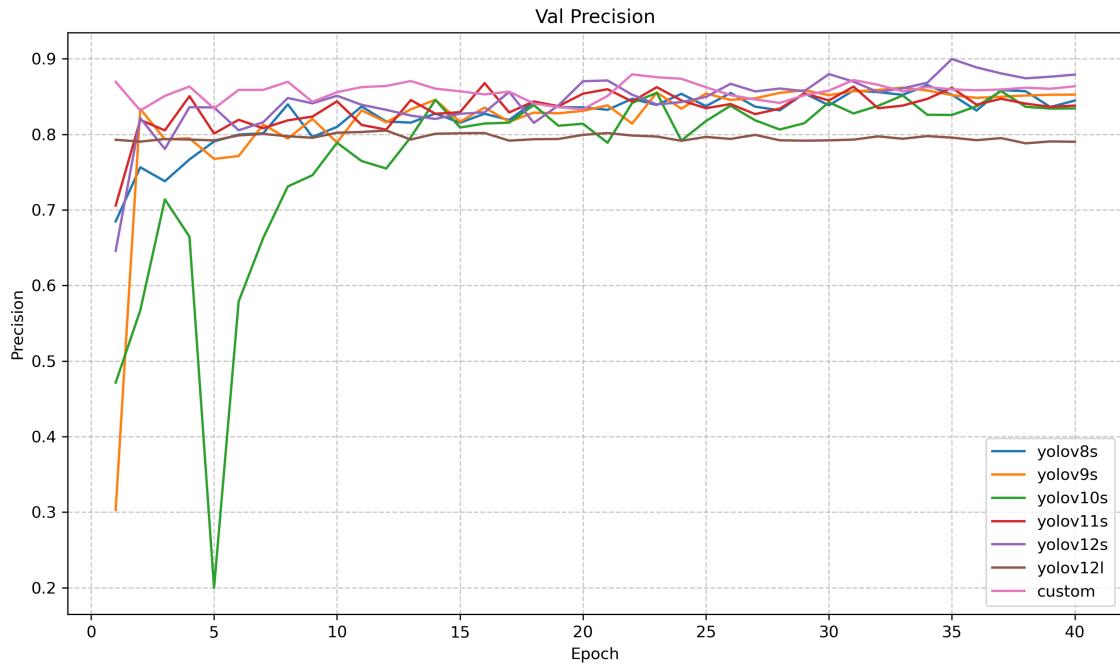


Figura 5.3: *Precision* del modelo sobre el conjunto de validación.

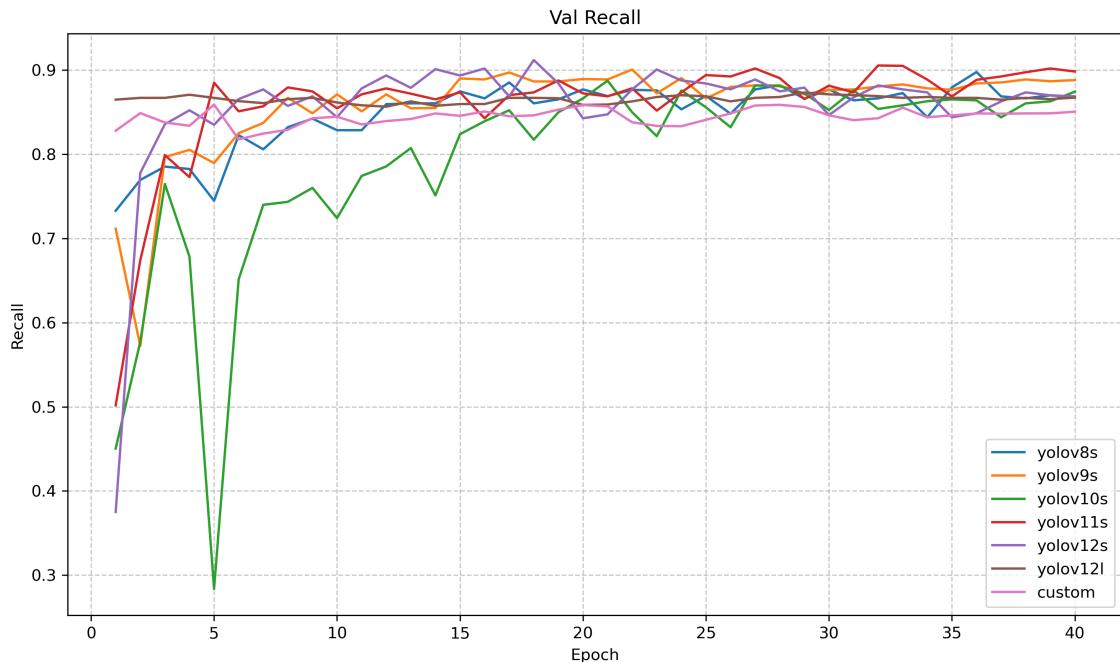


Figura 5.4: *Recall* del modelo sobre el conjunto de validación.

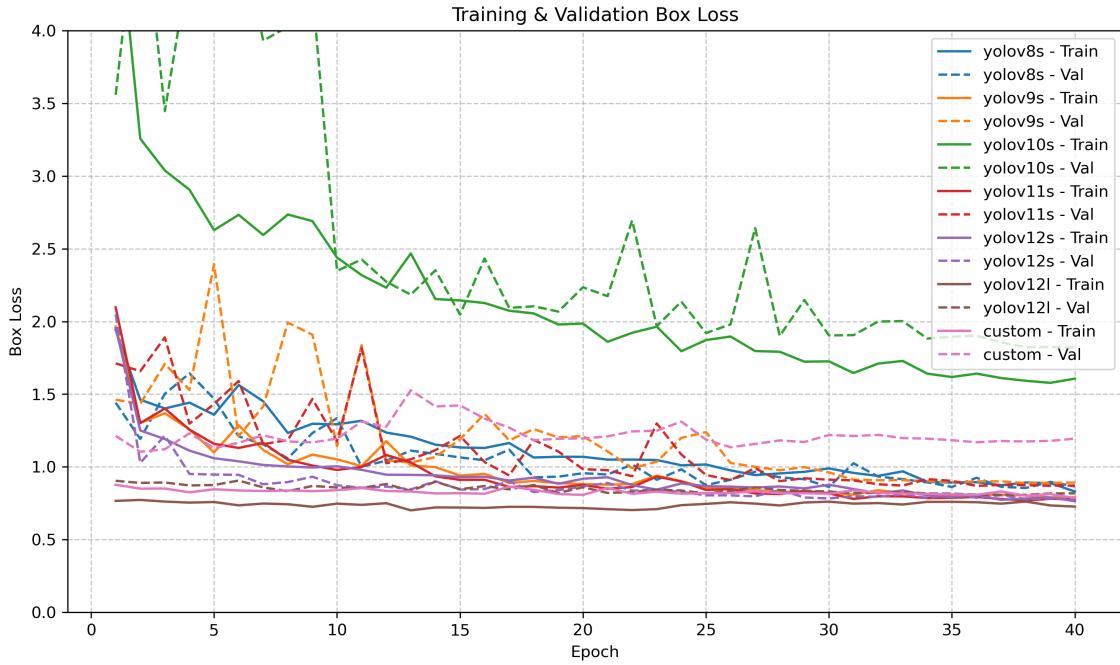


Figura 5.5: *Box Loss* sobre el conjunto de entrenamiento y validación.

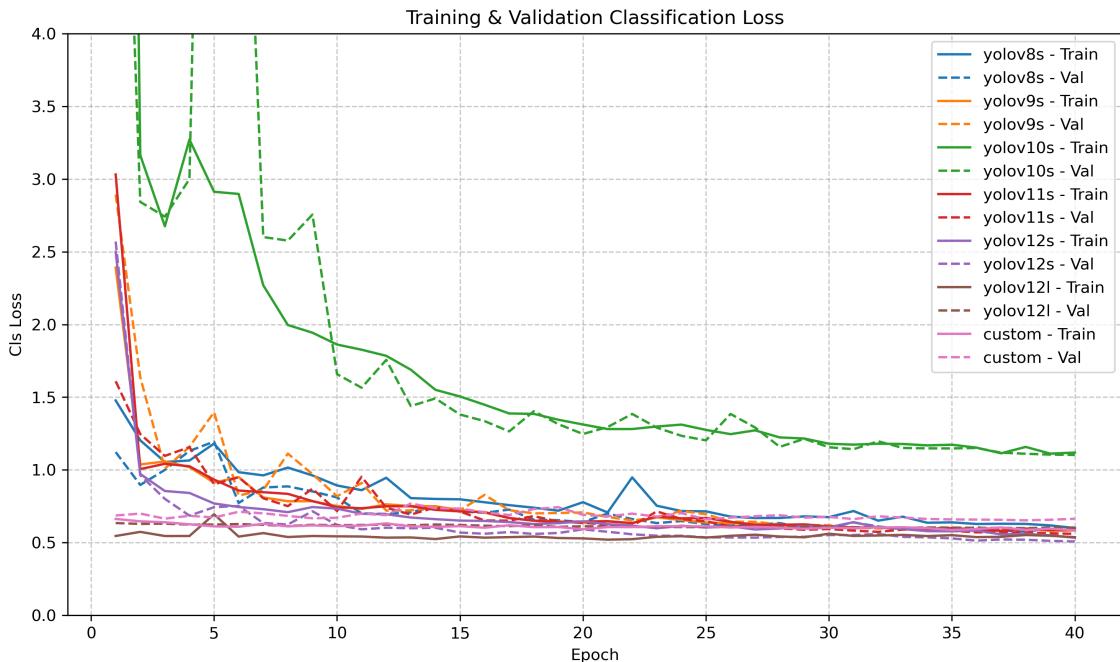


Figura 5.6: *Classification Loss* sobre el conjunto de entrenamiento y validación.

En general, todos los modelos muestran un buen desempeño durante el entrenamiento, con métricas que mejoran y se estabilizan a lo largo de las 40 épocas (*epoch*). Además, no hay evidencia de un sobreajuste (*overfitting*).

Aparentemente el modelo YOLOv12s arroja mejores resultados; principalmente, en la métrica más exigente, la mAP@0.5:0.95 (Figura 5.1). En base a esta métrica, el peor modelo se puede considerar el *custom*. Sorprendentemente, el modelo YOLOv12l (el más grande y preciso) es el peor modelo en cuanto a las métricas mAP@0.5 (Figura 5.2) y *precision* sobre el conjunto de validación (Figura 5.3).

A pesar de los buenos rendimientos del modelo YOLOv10s en las métricas de *precision*, *recall* y *mAP*, se aprecia que la función de pérdida (*loss*), tanto en la clasificación como en las *bounding boxes*, el modelo se distancia respecto al resto con una pérdida superior a la de los demás. A pesar de que no es una pérdida muy significativa, es importante tenerla en cuenta, por ser la métrica a minimizar durante el entrenamiento. Se puede ver que, en las primeras épocas, al modelo le cuesta mucho aprender a localizar y clasificar los objetos correctamente, aunque finalmente consigue estabilizarse como era de esperar.

5.3.2. Validación cruzada

Para demostrar la capacidad de generalización y fiabilidad de los diferentes modelos de detección de células redondas, se emplea la metodología de validación cruzada (*k-fold cross validation*). Para ello, se ha dividido el conjunto de entrenamiento original constituido por 373 imágenes (Tabla 5.1) en $k = 4$ pliegues o *folds* de tamaño similar. De este modo, el proceso de entrenamiento tiene lugar con las 4 combinaciones (sin repetición) posibles, en las que un *fold* se reserva para validar y tres para entrenar. Los resultados obtenidos para cada modelo se recogen en la Tabla 5.6.

Tabla 5.6: Resultados de la validación cruzada (*k-fold*) para todos los modelos

Modelo	Precision	Recall	mAP50	mAP50-95
YOLOv8s	$0,836 \pm 0,008$	$0,868 \pm 0,015$	$0,914 \pm 0,011$	$0,723 \pm 0,024$
YOLOv9s	$0,852 \pm 0,015$	$0,866 \pm 0,013$	$0,919 \pm 0,009$	$0,729 \pm 0,026$
YOLOv10s	$0,854 \pm 0,014$	$0,855 \pm 0,011$	$0,918 \pm 0,009$	$0,739 \pm 0,029$
YOLOv11s	$0,845 \pm 0,025$	$0,869 \pm 0,013$	$0,922 \pm 0,012$	$0,742 \pm 0,025$
YOLOv12s	$0,853 \pm 0,011$	$0,861 \pm 0,016$	$0,918 \pm 0,008$	$0,730 \pm 0,018$
YOLOv12l	$0,841 \pm 0,002$	$0,867 \pm 0,013$	$0,912 \pm 0,011$	$0,724 \pm 0,013$
custom	$0,836 \pm 0,014$	$0,865 \pm 0,014$	$0,905 \pm 0,011$	$0,707 \pm 0,015$

Los datos de la Tabla 5.6 se representan en la Figura 5.7. En esta figura, cada punto central indica el valor medio de rendimiento para una métrica específica

(precision, recall, mAP50 y mAP50-95), mientras que las barras horizontales que lo rodean representan la desviación estándar obtenida a lo largo de los 4 pliegues de la validación cruzada.

Analizando la Figura 5.7, se puede observar que, pese a las pequeñas diferencias en los promedios, los intervalos de incertidumbre se solapan entre todos los modelos para todas las métricas específicas a evaluar. Es un claro indicador de que los modelos ofrecen un rendimiento similar y robusto, que les permite generalizar de forma adecuada, pese al despunte del modelo YOLOv11s. Asimismo, las barras de incertidumbre más pequeñas reflejan qué modelos son más estables, destacando los modelos YOLOv12s, YOLOv12l y *custom*.

De este modo, la elección de un único "mejor" modelo es difícil, ya que son todos ellos opciones excelentes y estadísticamente equivalentes para esta tarea de detección de objetos.

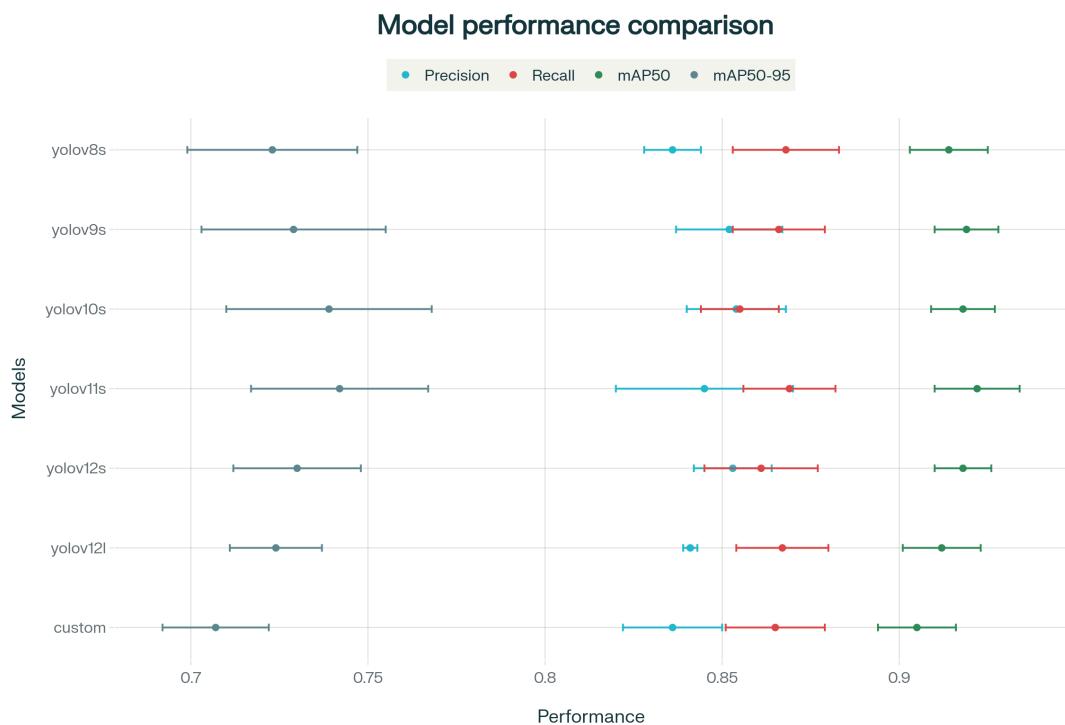


Figura 5.7: Representación de la validación cruzada (k-fold) para todos los modelos.

5.3.3. Ensamblado de modelos

Para mejorar la capacidad de generalización y el rendimiento de la detección de células redondas, se implementa una estrategia de ensamblado de modelos mediante la técnica de *Weighted Boxes Fusion (WBF)* [27].

Tras analizar los primeros resultados cuantitativos y cualitativos, se seleccionan dos modelos individuales con diferentes pesos, para la construcción del *ensemble*. YOLOv12 con sus capacidades para capturar información contextual se pondera con un peso de 0,6 mientras que YOLOv10s, que destaca por su eficiencia y precisión en la localización, se pondera con un peso de 0,4.

Asimismo, el proceso de ensamblado se realiza siguiendo los siguientes pasos:

- Para cada imagen, se obtienen las predicciones de los modelos individualmente.
- Se normalizan las coordenadas de las cajas para garantizar compatibilidad.
- Se aplica el algoritmo de WBF que agrupa *bounding boxes* similares de diferentes modelos y las fusiona mediante un promedio ponderado según: el peso asignado a cada modelo y la confianza de cada una de las predicciones individuales.
- Se aplica un umbral de confianza para eliminar aquellas predicciones con baja probabilidad.

Cabe destacar que, cuando las predicciones de YOLOv12 y YOLOv10 no coinciden (sin solapamiento significativo de las *bounding boxes*), el algoritmo mantiene las detecciones como predicciones independientes, aplica el umbral de omisión de cajas a cada predicción individual y si supera dicho umbral, estas predicciones se incluyen en el resultado final.

De este modo, se aprovechan las fortalezas de cada modelo y se reducen los falsos negativos al mantener detecciones únicas de cada modelo (mejora el *recall*).

Para obtener la configuración óptima de hiperparámetros, se exploran diferentes valores en un *grid*. Para el umbral de IoU (cuando dos *bounding boxes* se consideran superposiciones de la misma detección) se exploran los valores de 0,3, 0,4 y 0,5 y para el umbral de omisión de cajas (determina la confianza mínima para considerar una predicción) se exploran los valores de 0,25, 0,3 y 0,35.

La configuración óptima resultante fue un umbral de IoU de 0,5, un umbral de omisión de cajas de 0.35 y un umbral de IoU para cada modelo de 0.25.

5.3.4. Modelo personalizado

Este modelo personalizado o *custom* representa una arquitectura híbrida al combinar elementos específicos de tres generaciones de YOLOv8, YOLOv9 y YOLOv10 [27].

La arquitectura sigue el patrón típico de los modelos YOLO con una profundidad (controla el número de capas repetidas en la arquitectura) de 0,33 y un factor de anchura (controla el número de canales (filtros) en cada capa convolucional de la red) de 0,50, dando lugar a un equilibrio entre rendimiento y velocidad.

La arquitectura del modelo se segmenta en dos componentes clave:

Backbone

Este componente es el responsable de la extracción de características e integra las siguientes capas:

Tabla 5.7: Estructura del *backbone* del modelo personalizado

Índice	Tipo	Canales	Stride	Función
0-1	Conv	64/128	2/2	Extracción inicial de características
2	C2f (YOLOv8)	128	1	Agregación eficiente de características a baja escala
3-4	Conv + C2f	256	2	Captura de patrones a escala media
5-6	SCDown (YOLOv10) + C2f	512	2	Preservación de información espacial
7	SCDown (YOLOv10)	1024	2	Reducción contextual avanzada
8	RepNCSPELAN4 (YOLOv9)	[1024,512,256,1]	1	Reparametrización para patrones complejos
9	PSA	1024	1	Atención espacial píxel a píxel

Head

Responsable de interpretar las características y realizar la detección. Estructura FPN+PAN (*Feature Pyramid Network + Path Aggregation Network*):

Tabla 5.8: Estructura del *head* del modelo personalizado

Índice	Tipo	Canales	Función
10-12	Upsample + Concat + C2f	512	Ampliación y fusión con características P4
13-15	Upsample + Concat + C2f	256	Salida P3 para objetos pequeños
16-18	Conv + Concat + C2f	512	Salida P4 para objetos medianos
19-21	Conv + Concat + C2f	1024	Salida P5 para objetos grandes
22	Detect	[nc]	Detección final a múltiples escalas

Las características de este modelo son:

- Uso de SCDown de YOLOv10 que preserva mejor la información espacial durante la reducción dimensional.
- Integración del módulo RepNCSPELAN4 de YOLOv9 para mejorar la capacidad de representación.
- Incorporación de atención espacial (PSA) para enfocarse en regiones relevantes.
- Estructura FPN+PAN optimizada para la detección a múltiples escalas.

5.4. Métricas

Para la evaluación de los modelos, se emplea un conjunto de métricas fundamentales que se exponen a continuación.

Precision : Mide la proporción de detecciones correctas entre todas las detecciones realizadas:

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

donde TP son los verdaderos positivos y FP los falsos positivos. Alta *precision* indica pocas detecciones erróneas.

Recall : Mide la proporción de instancias reales que han sido detectadas:

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

donde FN son los falsos negativos. Un *recall* alto indica que el modelo encuentra la mayoría de las instancias reales.

F1-score : Es la media armónica entre la precisión y el *recall*, proporcionando una medida única que equilibra ambos aspectos. Se calcula como:

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

El *F1-score* es especialmente útil cuando es importante considerar tanto los falsos positivos como los falsos negativos. En este trabajo se utiliza como métrica principal en las curvas F1-confianza y en la herramienta web.

mean Average Precision (mAP) : Para detección de objetos se calcula la curva *precision-recall* para cada clase y su área bajo la curva (AP). El *mean Average*

Precision es la media de las AP sobre todas las clases:

$$\text{mAP} = \frac{1}{C} \sum_{c=1}^C \text{AP}_c$$

donde C es el número de clases. En detección se considera una predicción como verdadero positivo si el *Intersection over Union* (IoU) entre la caja predicha y la caja *ground truth* supera un umbral (por ejemplo $\text{IoU} \geq 0.5$).

mAP@0.5 (mAP50): Es la mAP calculada usando un único umbral de IoU igual a 0.5. Formalmente:

$$\text{mAP}@0.5 = \frac{1}{C} \sum_{c=1}^C \text{AP}_c(\text{IoU} = 0.5)$$

Es una medida menos exigente, que acepta solapamientos moderados entre predicción y *ground truth*.

mAP@[0.5:0.95] (mAP50:95): Es la mAP estándar del *benchmark* COCO que promedia la AP de cada clase sobre múltiples umbrales de IoU desde 0.50 hasta 0.95 con paso 0.05 (10 umbrales: 0.50, 0.55, ..., 0.95). Nota: COCO calcula cada AP integrando la curva *precision-recall* muestrada en 101 puntos, y después se promedian las AP en los 10 umbrales para obtener mAP@[0.5:0.95].

$$\text{mAP}_{[0.5:0.95]} = \frac{1}{C} \frac{1}{T} \sum_{c=1}^C \sum_{t \in \{0.50, 0.55, \dots, 0.95\}} \text{AP}_c(\text{IoU} = t)$$

donde $T = 10$. Esta métrica es más exigente porque penaliza detecciones con IoU bajos y refleja mejor la precisión espacial del modelo.

Inferencia (ms) Tiempo medio (en milisegundos) que tarda un modelo en procesar una única imagen (inferencia). Relevante para proyectos en tiempo real.

6. Resultados

En este capítulo se abordan los resultados obtenidos para los modelos sobre los diferentes conjuntos de *test*. Para esto, se analizan los resultados atendiendo a dos enfoques complementarios: por un lado, se presentan los resultados cuantitativos, basados en métricas objetivas de evaluación; y por otro, se realiza un análisis cualitativo, centrado en la interpretación visual y contextual de las predicciones realizadas por los modelos.

6.1. Resultados cuantitativos

En esta primera parte, se evalúan los resultados sobre el conjunto de *test* original para el cual se tienen resultados asociados a la prueba de concepto previa (Tabla 6.1), bajo la premisa de que todos los modelos arrojan resultados similares, por los tiempos de inferencia y el valor más alto en la métrica mAP@0.5:0.95, el mejor modelo es YOLOv7.

Tabla 6.1: Resultados de los modelos en la prueba de concepto inicial

Modelo	Precisión (P)	recall (R)	mAP@0.5	mAP@0.5:0.95	Inferencia (ms)
YOLOv7	0,870	0,889	0,935	0,722	20,0
YOLOv7-W6	0,847	0,883	0,925	0,694	16,0
YOLOv7-E6E	0,906	0,857	0,934	0,721	127,5

Tabla 6.2: Resultados de los modelos sobre el conjunto de *test* original

Modelo	Precisión (P)	recall (R)	mAP@0.5	mAP@0.5:0.95	Inferencia (ms)
YOLOv8s	0,843	0,844	0,920	0,722	11,006
YOLOv9s	0,838	0,852	0,921	0,721	13,485
YOLOv10s	0,821	0,867	0,918	0,717	11,221
YOLOv11s	0,861	0,824	0,921	0,732	11,248
YOLOv12s	0,856	0,863	0,936	0,760	14,615
YOLOv12l	0,856	0,836	0,921	0,729	48,146
<i>custom</i>	0,835	0,838	0,909	0,695	13,058
<i>ensemble</i>	0,815	0,895	0,864	0,697	75,073

Comparando estos resultados con los obtenidos en este proyecto (Tabla 6.2), el modelo YOLOv12s destaca por mejorar las métricas mAP@0.5 y mAP@0.5:0.95 (un 5.26 % más que YOLOv7) y reducir la latencia respecto a versiones anteriores, por lo que se considera el mejor modelo en esta prueba. Aunque los modelos más pesados como YOLOv12l y el ensamblado presentan mayores tiempos de inferencia, todos los modelos mantienen una latencia inferior a 100 ms. El *ensemble* logra el mayor *recall* (0.895), pero con peores valores de mAP, precisión y latencia. YOLOv8s es el más rápido (11.006 ms) y YOLOv11s obtiene la mayor precisión.

Analizando los nuevos conjuntos de *test* (Tabla 6.3), los resultados siguen siendo similares entre modelos. Según la métrica mAP@0.5:0.95, YOLOv12s vuelve a ser el mejor (0.673), seguido de YOLOv12l (0.659). El *ensemble* mejora precisión y *recall*, pero empeora mAP y latencia. El modelo *custom* destaca por su baja latencia y buenos resultados, especialmente en test3, donde iguala la precisión del *ensemble* (0.873). Las diferencias entre el resto de modelos son mínimas.

Tras el proceso de reetiquetado del conjunto de *test* original, es decir, en la evaluación sobre el conjunto de *test*, se observan cambios interesantes que sugieren una mejora en la calidad del *ground truth*. La mayoría de los modelos muestran una mejora en la *precision*, destaca el modelo *ensemble* con el mayor incremento (+0,054). En cuanto al *recall*, los resultados son más variados, pero YOLOv12l y YOLOv11s muestran mejoras significativas. La métrica mAP@0.5 mejora en la mayoría de los modelos mientras que la métrica mAP@0.5:0.95 sufre un descenso generalizado. Esto sugiere que, si bien los modelos identifican mejor las células redondas, la precisión de las *bounding boxes* ha disminuido ligeramente. Asimismo, los tiempos de inferencia se mantienen con variaciones mínimas en la mayoría de los modelos.

El conjunto test2 muestra latencias más altas (31–76 ms) y un descenso general de mAP@0.5:0.95, lo que indica condiciones diferentes que afectan al rendimiento.

En resumen, YOLOv12s es el modelo más destacado entre las arquitecturas evaluadas. El ensamblado de modelos mejora *recall* y precisión al reducir falsos negativos, aunque aumenta los falsos positivos, como muestran las matrices de confusión (Sección D.1). En test2, las diferencias son mínimas debido al tamaño reducido del conjunto. El análisis de las curvas F1-confianza (Sección D.2) indica que el *F1-score* óptimo se alcanza con umbrales de confianza entre 0.25 y 0.5, por lo que se estudia YOLOv12s con un umbral de 0.35 para imágenes de alta resolución.

Tabla 6.3: Evaluación de modelos sobre los nuevos conjuntos de *test*

Modelo	Test	Precisión	Recall	mAP@0.5	mAP@0.5:0.95	Inferencia (ms)
YOLOv8s	test	0,855	0,838	0,926	0,704	11,844
YOLOv9s	test	0,851	0,876	0,933	0,706	13,492
YOLOv10s	test	0,851	0,862	0,927	0,701	11,120
YOLOv11s	test	0,846	0,866	0,930	0,716	12,143
YOLOv12s	test	0,862	0,848	0,936	0,740	14,519
YOLOv12l	test	0,846	0,889	0,934	0,714	45,214
<i>custom</i>	test	0,845	0,853	0,919	0,678	12,913
<i>ensemble</i>	test	0,869	0,860	0,842	0,664	69,572
YOLOv8s	test2	0,924	0,938	0,969	0,553	30,901
YOLOv9s	test2	0,917	0,927	0,961	0,542	46,502
YOLOv10s	test2	0,956	0,898	0,962	0,589	34,355
YOLOv11s	test2	0,929	0,917	0,964	0,534	33,114
YOLOv12s	test2	0,965	0,917	0,975	0,556	51,508
YOLOv12l	test2	0,957	0,926	0,975	0,562	75,871
<i>custom</i>	test2	0,945	0,903	0,957	0,539	35,626
<i>ensemble</i>	test2	0,919	0,944	0,935	0,509	54,958
YOLOv8s	test3	0,842	0,862	0,937	0,700	12,815
YOLOv9s	test3	0,843	0,870	0,935	0,680	14,772
YOLOv10s	test3	0,873	0,826	0,926	0,668	11,230
YOLOv11s	test3	0,855	0,874	0,935	0,698	10,875
YOLOv12s	test3	0,836	0,821	0,922	0,723	15,439
YOLOv12l	test3	0,849	0,894	0,942	0,701	47,256
<i>custom</i>	test3	0,873	0,843	0,928	0,660	11,373
<i>ensemble</i>	test3	0,873	0,857	0,837	0,663	63,758

6.2. Resultados cualitativos

Uno de los problemas más comunes en la adquisición de imágenes microscópicas es la presencia de artefactos (anillos de Newton), que pueden interferir con la detección precisa de células redondas. Por esta razón, se analizan los diferentes modelos sobre un conjunto de imágenes que contienen estos artefactos. Para pasar la evaluación, se establece que el modelo no debe detectar ningún Anillo de Newton como célula redonda. Para pasar la evaluación, se establece que el modelo no debe detectar ningún Anillo de Newton como célula redonda, por lo que se ha fijado un umbral de confianza alto (0,5 y 0,7) para minimizar las detecciones erróneas. Los resultados obtenidos se muestran en la Tabla 6.4.

Tabla 6.4: Evaluación de modelos sobre imágenes con artefactos

Modelo/Umbral	59 imagen	61 imagen	219 imagen	369 imagen	already tested-00	already tested-15	already tested-16	already tested-17	already tested-22
Test	1	1	1	1	3	3	3	3	3
YOLOv8s 0,5	x	x	x	✓	✓	✓	✓	✓	✓
YOLOv8s 0,7	✓	✓	✓	✓	✓	✓	✓	✓	✓
YOLOv9s 0,5	x	x	x	x	✓	x	x	✓	✓
YOLOv9s 0,7	✓	x	x	x	✓	x	✓	✓	✓
YOLOv10s 0,5	x	✓	✓	✓	✓	✓	✓	✓	x
YOLOv10s 0,7	✓	✓	✓	✓	✓	✓	✓	✓	✓
YOLOv11s 0,5	✓	x	x	x	✓	x	✓	✓	✓
YOLOv11s 0,7	✓	✓	✓	✓	✓	✓	✓	✓	✓
YOLOv12s 0,5	✓	✓	✓	✓	✓	✓	✓	✓	✓
YOLOv12s 0,7	✓	✓	✓	✓	✓	✓	✓	✓	✓
YOLOv12l 0,5	x	x	x	x	✓	x	x	✓	x
YOLOv12l 0,7	x	x	x	x	✓	x	x	✓	x
custom 0,5	x	✓	x	x	✓	✓	x	✓	x
custom 0,7	✓	✓	✓	x	✓	✓	✓	✓	✓
ensemble	✓	✓	✓	x	✓	✓	✓	✓	✓

Tres modelos destacan considerablemente frente al resto: YOLOv10s, YOLOv12s y el *ensemble*; los dos primeros motivaron la construcción del ensamblado descrito en Subsección 5.3.3, ofreciendo un equilibrio competitivo. Destaca la robustez de YOLOv12s, que no detecta ningún artefacto como célula redonda, incluso con un umbral de confianza bajo (0.5).

Para ampliar la comparativa visual, se seleccionan imágenes representativas del conjunto de test y se muestran las predicciones de YOLOv12s (con un umbral de confianza de 0.5 y 0.25) y el ensamblado junto con el *ground truth* de los expertos y la salida del modelo de la prueba de concepto (YOLOv7).

Con el fin de garantizar una comparación justa con la prueba de concepto inicial, que operaba a probabilidad 0.25, se fijó el mismo umbral (0.25) para YOLOv12s; en las imágenes de la Sección D.3, tanto YOLOv12s (con umbral de 0.5) como el *ensemble* se aproximan al *ground truth* pero presentan muchos falsos negativos. Por el contrario, YOLOv7 presenta pocos falsos negativos y muchos falsos positivos. Sin

embargo, con un umbral de 0.25, YOLOv12s consigue un buen equilibrio al presentar pocos falsos positivos (bastantes menos que YOLOv7) y pocos falsos negativos, dando lugar a una detección más precisa y fiable.

En conjunto, YOLOv12s se perfila como el modelo más adecuado para la detección de células redondas en este *dataset* y, junto con la herramienta web, constituye una solución operativa para análisis automatizado; además, el ajuste interactivo del umbral de confianza permite adaptar sensibilidad y especificidad a las necesidades de cada caso.

7. Herramienta Web

Con el propósito de definir una herramienta de análisis automatizado, se desarrolla una interfaz intuitiva e interactiva para la detección de células redondas sobre una muestra individual o colectiva. La información relativa a la aplicación, se puede encontrar en el manual de usuario (Sección C.1).

7.1. Objetivos de la herramienta

La herramienta web para la detección de células redondas tiene como principales objetivos:

- **Detección automática de células en imágenes microscópicas en el contexto médico abordado con anterioridad:** Permitir identificar y localizar células redondas dotando al usuario de diferentes arquitecturas de modelos.
- **Comparación de rendimiento entre modelos:** Facilitar la evaluación comparativa entre diferentes modelos, permitiendo al usuario seleccionar el mejor modelo y el intervalo de actuación para el IoU.
- **Visualización de resultados:** Proporcionar una interfaz de visualización clara e intuitiva que permita al usuario visualizar las predicciones del modelo y las *bounding boxes* del *ground truth* si se dispone de las mismas.
- **Análisis cuantitativo:** Permitir obtener métricas y estadísticas como: *precision*, *accuracy*, conteo o IoU promedio.
- **Soporte a la investigación biomédica:** Facilitar la identificación temprana de diferentes grados de patología.

7.2. Arquitectura

La arquitectura de la aplicación sigue un diseño modular y escalable, basada en *Streamlit*. Esto permite tener una aplicación monolítica que encapsula toda la

funcionalidad en un único ejecutable y mantiene una clara separación entre la lógica de procesamiento y de presentación.

7.2.1. Organización del proyecto

La estructura principal es:

- `app.py`: orquestación de la interfaz (*widgets Streamlit*), gestión de sesión y del flujo de inferencia.
- `utils/utils.py`: servicios de *backend* lógico (carga de modelos, preprocesado, inferencia, postprocesado, visualización).
- `assets/styles.css`: estilo e integración visual mediante `st.markdown`.
- `models/{final_model_yolovXs}/`: modelos entrenados para la casuística.
- `runs/detect/`: salidas temporales de inferencia (imágenes anotadas, JSON/CSV).

Para más información, revisar la documentación `04.Code/cell_detection_App` en el repositorio del proyecto [27].

7.2.2. Capas funcionales en *Streamlit*

1. **Interfaz y orquestación (UI)**: componentes como `st.file_uploader`, `st.selectbox`, `st.slider`, `st.sidebar` y `st.image` permiten la interacción con el usuario de una forma intuitiva.
2. **Servicios de modelo e inferencia**: Funciones en `utils/utils.py` para cargar pesos YOLO, seleccionar dispositivo (CPU/GPU), normalizar entradas y ejecutar el modelo sobre las entradas preprocesadas.
3. **Pipeline de datos**: Preprocesado (lectura, redimensionado, normalización), postprocesado (NMS, filtrado por confianza, conversión a anotaciones PASCAL VOC), y renderizado de *bounding boxes*. Además, incluye la exportación de las predicciones en formato PASCAL VOC.
4. **Estado y caché**: `st.session_state` para parámetros y resultados interactivos; `@st.cache_resource` evita recargar pesos al cambiar sólo parámetros de inferencia; `@st.cache_data` para memoizar resultados derivados (tablas/figuras).

5. **Estilos y experiencia de usuario:** `assets/styles.css` para mejorar el aspecto visual que por defecto ofrece *Streamlit* y uso de *layout* responsivo (columnas, botones, *tooltips*) para mejorar la usabilidad.
6. **Manejo de errores:** `st.info` para informar de estados (dispositivo, imágenes y xml subidos), `st.warning` para advertir de un error en la lectura del xml, `st.error` impide continuar con la ejecución si no encuentra modelo o al no ser capaz de procesar una imagen.

7.2.3. Flujo de ejecución

1. Inicio: se inicializa la sesión y se aplica el estilo personalizado CSS.
2. El usuario selecciona modelo e IoU: YOLOv12s y 0,5.
3. Carga de modelo (una sola vez) vía `@st.cache_resource`.
4. Carga de imagen (individual o lote) con `st.file_uploader`. Opcional: cargar el xml con las anotaciones en formato PASCAL VOC.
5. Preprocesado de la imagen y envío al dispositivo (CPU/GPU).
6. Inferencia YOLO y postprocesado (NMS, métricas básicas).
7. Visualización: imagen anotada, conteo, tablas, métricas y *bounding boxes*.

8. Conclusión

En este capítulo se abordan las conclusiones generales del proyecto, las limitaciones que dificultan el desarrollo del mismo y las líneas de trabajo a futuro.

8.1. Conclusión general

Este Trabajo de Fin de Máster cumple con éxito su objetivo principal: el diseño, desarrollo y validación de un sistema inteligente para la detección automatizada de células redondas en imágenes de muestras de semen, demuestra la viabilidad de integrar soluciones de aprendizaje profundo en un flujo de trabajo de análisis clínico.

Se valida la hipótesis inicial, ya que la exploración de diferentes arquitecturas de la familia YOLO, junto con la optimización de hiperparámetros, permite superar el rendimiento de la prueba de concepto inicial. El modelo YOLOv12s demuestra robustez y generalización, no solo mejorando métricas cuantitativas clave como mAP@0.5:0.95, sino también en los resultados cualitativos, tanto en la no detección de anillos de Newton bajo un intervalo de confianza aceptable sino también en la reducción de el número de FP y FN en detecciones con umbrales de confianza mucho más sensibles.

Un aspecto a destacar es el proceso de reetiquetado con *feedback* de expertos. Este proceso sirve para tener una visión más realista del rendimiento de los modelos, destaca una mejoría generalizada en el conjunto de métricas pero también revela una mayor dificultad para alcanzar una localización perfecta (mAP@0.5:0.95). Esta observación pone de manifiesto la complejidad inherente del problema y la importancia de contar con datos de alta calidad para una evaluación rigurosa del rendimiento.

Este trabajo no se limita al desarrollo de modelos de detección de células sino que va un paso más y los integra de una manera interactiva mediante una herramienta web. De este modo, no solo se facilita la validación de los modelos, sino que se establece un puente hacia su futura integración en un entorno de laboratorio real.

En definitiva, el proyecto consigue dar una solución que va desde el preprocesamiento de datos y el entrenamiento de diferentes modelos hasta el despliegue de una aplicación funcional. Se demuestra así que, es posible automatizar una tarea

tan compleja y subjetiva como es la detección de células redondas; sentando así, las bases para optimizar el espermograma y potenciar la transformación digital en el campo de la andrología.

8.2. Limitaciones del estudio

A pesar de los buenos resultados, es importante reconocer las limitaciones inherentes al trabajo:

- **Subjetividad y ambigüedad en las anotaciones:** La principal limitación del proyecto radica en la propia naturaleza del problema. Como se documenta en la literatura, la identificación de células redondas en muestras de semen es una tarea compleja y sujeta a una alta variabilidad, incluso entre expertos. Autores como Chen admiten que, en ciertos casos, es difícil distinguir una impureza de un espermatozoide, lo que introduce una ambigüedad inevitable en los datos de entrenamiento que perjudica el aprendizaje del modelo [14]. Esta ambigüedad en el *ground truth* supone una limitación en el rendimiento máximo a alcanzar por cualquier modelo.
- **Heterogeneidad de la clase "célula redonda":** Las "células redondas" no se abordan como una única categoría sino que agrupa un conjunto heterogéneo de diferentes células. Esto simplifica el problema de detección pero se pierde información valiosa que una clasificación más detallada podría proporcionar.
- **Alcance y representatividad del dataset:** Aunque el conjunto de datos proporcionado por Microptic S.L. [2] es suficiente para demostrar la viabilidad de la prueba de concepto, el número de datos que lo conforman es bastante limitado si atendemos a los grandes volúmenes de datos que suelen abordar los modelos de aprendizaje profundo. Además, todos los datos provienen de la misma empresa, limitando así, la capacidad de generalizar de los modelos y la externalización del *software* a otros laboratorios.
- **Recursos computacionales:** Si bien durante el proceso de entrenamiento se emplea *hardware* limitado, el uso de un servidor, pese a las implicaciones económicas que conlleva, puede dar solución a estas limitaciones. La disponibilidad de mejores recursos computacionales permite no solo explorar nuevas arquitecturas sino realizar ajustes más profundos.

8.3. Lineas de trabajo futuro

Los resultados de este proyecto abren nuevas y prometedoras vías de investigación. Para continuar avanzando en la detección automatizada de células redondas se proponen las siguientes líneas de trabajo:

- **Exploración de otras arquitecturas:** Esta prueba de concepto se enfoca en detectores de una única etapa (YOLO), optimizados para ofrecer un equilibrio entre precisión y velocidad. Sin embargo, una alternativa a estos modelos pueden ser los de dos etapas (como *Fast-RCNN* [29]), el cual presenta generalmente mayor *precision* y tiempo de inferencia. Una alternativa a esta línea de investigación es usar detectores basados en Transformers (como D-FINE [30]). Este enfoque promete una localización muy precisa incluso en escenas complejas y podría superar algunas limitaciones de los modelos puramente convolucionales.
- **Gestión de los artefactos:** La baja representación de los anillos de Newton nos dicen que la introducción de una clase nueva puede continuar dando falsos positivos en las diferentes detecciones de clases, dando lugar a nuevos problemas. Sin embargo, es una vía a explorar que no se puede descartar. Esta hipótesis es necesaria si se plantea un grado superior de automatización, donde la precisión es relevante, y por tanto, el umbral de confianza debe ser menor.
- **Enriquecimiento del *dataset*:** Es bien sabido que la robustez de los modelos depende de la calidad y diversidad de los datos de entrenamiento. Es fundamental continuar ampliando el *dataset* con imágenes procedentes de diferentes condiciones. Un conjunto de datos más extenso y heterogéneo no solo mejoraría la capacidad de generalización de los modelos sino que abre las puertas a la detección de otras células.
- **Evolución y validación de la herramienta web:** La evolución de la herramienta a su vez debe pasar por la inclusión de nuevos modelos como *Fast-RCNN*. El siguiente paso lógico es desplegar la herramienta web en un entorno de laboratorio real para obtener el *feedback* directo de los expertos. Se puede implementar una funcionalidad que permita a los especialistas corregir las predicciones del modelo, creando un ciclo de aprendizaje activo donde sus validaciones se utilicen para reentrenar y mejorar continuamente la precisión del sistema.

Bibliografía

- [1] Manual de laboratorio de la OMS para el examen y procesamiento del semen humano, sexta edición [WHO laboratory manual for the examination and processing of human semen, sixth edition]. Ginebra: Organización Mundial de la Salud, 2025.
- [2] Microptic S.L. <https://www.micropticsl.com/>.
- [3] Hamilton Thorne. Advanced reproductive technologies. Disponible en: <https://www.hamiltonthorne.com/>, 2024.
- [4] DIGIS3, Digitalización Inteligente, Sostenible y Cohesiva. <https://digis3.eu/es>.
- [5] Ultralytics. Ultralytics models documentation. <https://docs.ultralytics.com/es/models/>.
- [6] Hamilton Thorne. Relevance of round cells detection in spermograms. Disponible en: <https://www.hamiltonthorne.com/relevance-of-round-cells-detection-in-spermograms/>, 2024.
- [7] Ranjan Sapkota, Zhichao Meng, Martin Churuvija, Xiaoqiang Du, Zenghong Ma, and Manoj Karkee. Comprehensive performance evaluation of yolov12, yolov11, yolov10, yolov9 and yolov8 on detecting and counting fruitlet in complex orchard environments. *arXiv preprint arXiv:2407.12040v7*, 2025.
- [8] Optuna Development Team. Optuna: Automl and hyperparameter optimization framework. <https://github.com/optuna/optuna>, 2025. Accessed: 2025-08-17.
- [9] Manesh Kumar Panner Selvam, Ajaya Kumar Moharana, Saradha Baskaran, Renata Finelli, Matthew C. Hudnall, and Suresh C. Sikka. Current updates on involvement of artificial intelligence and machine learning in semen analysis. *Medicina*, 60(2), 2024.
- [10] Rania Maalej, Olfa Abdelkefi, and Salima Daoud. Advancements in automated sperm morphology analysis: a deep learning approach with comprehensive classification and model evaluation. *Multimedia Tools and Applications*, 84(23):27345–27378, 2025.

- [11] S. Long and S. Kenworthy. Round cells in diagnostic semen analysis: A guide for laboratories and clinicians. *British Journal of Biomedical Science*, Volume 79 - 2021, 2022.
- [12] E. Johansson, A. Campana, R. Luthi, and A. de Agostini. Evaluation of ‘round cells’ in semen analysis: a comparative study. *Human Reproduction Update*, 6(4):404–412, 07 2000.
- [13] Yujiao Sun, Shihao Shao, Jiangwei Huang, Hao Shi, Liying Yan, Yongjie Lu, Ping Liu, Yuqiang Jiang, Jie Qiao, and Li Zhang. Development and validation of a deep learning model based on cascade mask regional convolutional neural network to noninvasively and accurately identify human round spermatids. *Journal of Advanced Research*, 2025.
- [14] Ao Chen, Feng-Lei Fan, Jinghua Zhang, Md Mamunur Rahaman, Rui Li, Jiang Tao, Tieyong Zeng, Marcin Grzegorzek, and Chen Li. Active: A deep network for sperm and impurity detection in microscopic videos. In *2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 5247–5256, 2024.
- [15] ¿cuánto cobra un ingeniero informático en españa? <https://www.ilerna.es/blog/cuanto-cobra-ingeniero-informatico>.
- [16] Sueldo de scrum master en españa. https://www.glassdoor.com.ar/Sueldos/spain-scrum-master-sueldo-SRCH_IL.0,5_IN219_K06,18.htm.
- [17] E2e networks pricing. <https://www.e2enetworks.com/pricing>.
- [18] Github copilot. <https://github.com/features/copilot>.
- [19] Tipos impositivos del iva 2025 - agencia tributaria. https://sede.agenciatributaria.gob.es/static_files/Sede/Tema/IVA/IVA_reperc/Tipos_IVA_2025.pdf.
- [20] What is scrum? - scrum.org learning series. <https://www.scrum.org/learning-series/what-is-scrum/>.
- [21] Online gantt — gantt chart maker. <https://www.onlinegantt.com/>.
- [22] Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales. Boletín Oficial del Estado, núm. 294, Diciembre 2018. Disponible en: <https://www.boe.es/eli/es/lo/2018/12/05/3/con>.

- [23] Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos. Diario Oficial de la Unión Europea, Abril 2016. Disponible en: <https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=CELEX%3A32016R0679>.
- [24] Reglamento (UE) 2024/1689 del Parlamento Europeo y del Consejo, de 13 de junio de 2024, por el que se establecen normas armonizadas en materia de inteligencia artificial (Reglamento de Inteligencia Artificial). Diario Oficial de la Unión Europea, L 2024/1689, julio 2024. Entrada en vigor: 1 de agosto de 2024. Disponible en: <https://eur-lex.europa.eu/eli/reg/2024/1689/oj>.
- [25] Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual. Boletín Oficial del Estado, núm. 97, Abril 1996. Disponible en: <https://www.boe.es/buscar/act.php?id=BOE-A-1996-8930>.
- [26] labelImg: Graphical image annotation tool. [https://github.com/tzutalin\(labelImg](https://github.com/tzutalin(labelImg).
- [27] Aitor García Blanco. Repositorio del TFM: Detección de células defectuosas en imágenes médicas. <https://github.com/AigarciaabFabero/TFM>, 2025.
- [28] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 2623–2631, New York, NY, USA, 2019. Association for Computing Machinery.
- [29] Ross Girshick. Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.
- [30] Yansong Peng, Hebei Li, Peixi Wu, Yueyi Zhang, Xiaoyan Sun, and Feng Wu. D-fine: Redefine regression task in detrs as fine-grained distribution refinement, 2024.
- [31] Priya S. Patil, Rajendra S. Humbarwadi, Ashalata D. Patil, and Anita R. Gune. Immature germ cells in semen — Correlation with total sperm count and sperm motility. *Journal of Cytology*, 30(3):185–189, July 2013.

Anexo A

Control de versiones

En el marco de desarrollo del Trabajo de Fin de Máster (TFM), se emplea GitHub como servicio de control de versiones para gestionar eficientemente el código fuente y la documentación del proyecto; facilitando el seguimiento, la trazabilidad y la colaboración.

Esta herramienta permite un seguimiento del proyecto por parte del responsable o tutor. Además, de ser necesario, GitHub presenta una funcionalidad para desarrolladores que permite editar el proyecto desde un navegador web. Facilitando la implementación de mejoras y la corrección de errores desde cualquier dispositivo con acceso a internet.

Casi la totalidad del desarrollo de este proyecto final lo podemos encontrar en el repositorio personal [27], cumpliendo en todo momento, con la normativa y legislación bajo las que se enmarca el proyecto.

Anexo B

Seguimiento del proyecto

En este anexo se recoge el diagrama de Gantt que refleja la planificación temporal del proyecto, así como las tareas realizadas en cada una de las fases del mismo.

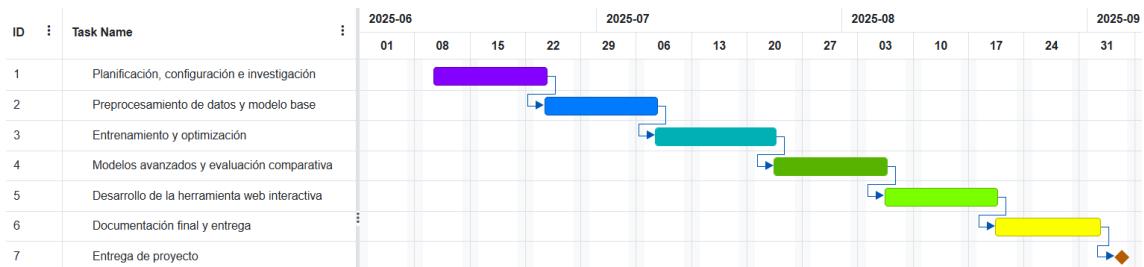


Figura B.1: Diagrama de Gantt para la planificación del proyecto

Anexo C

Herramienta Web

C.1. Manual de usuario

Para facilitar la interacción del usuario con la herramienta web vista con anterioridad, se define la siguiente guía de usuario.

Desde el entorno, con la dependencia *Streamlit*, se ejecuta el comando `streamlit run app.py` y se lanza la aplicación en el navegador predeterminado como se muestra en la siguiente imagen.

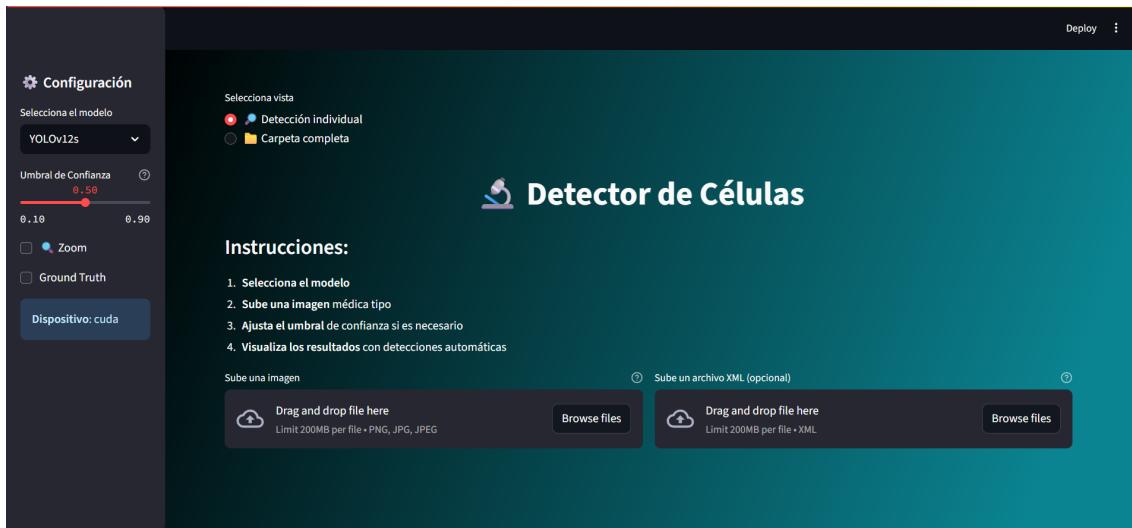


Figura C.1: Interfaz de la aplicación web.

En este punto, el usuario puede tomar la decisión de procesar una imagen individual o un conjunto de imágenes. Para esto, tenemos el *widget* que permite la selección entre "Detección individual" o "Carpeta completa". En ambas situaciones, la aplicación nos muestra una guía básica de funcionamiento en el apartado "Introducciones" en la Figura C.1. De manera totalmente opcional, el usuario puede cargar junto a la imagen a estudio, el correspondiente conjunto de anotaciones.

El *widget* de "Configuración" permite la selección manual del modelo a través de un desplazable, el umbral de confianza de IoU, la representación del *ground truth*

(siempre que exista el xml asociado a la imagen) mediante un *checkbox* (Figura C.2) y realizar un análisis exploratorio sobre la imagen con un zoom (Figura C.3). Asimismo, muestra el dispositivo (CPU o GPU) con el que se procesan las imágenes.

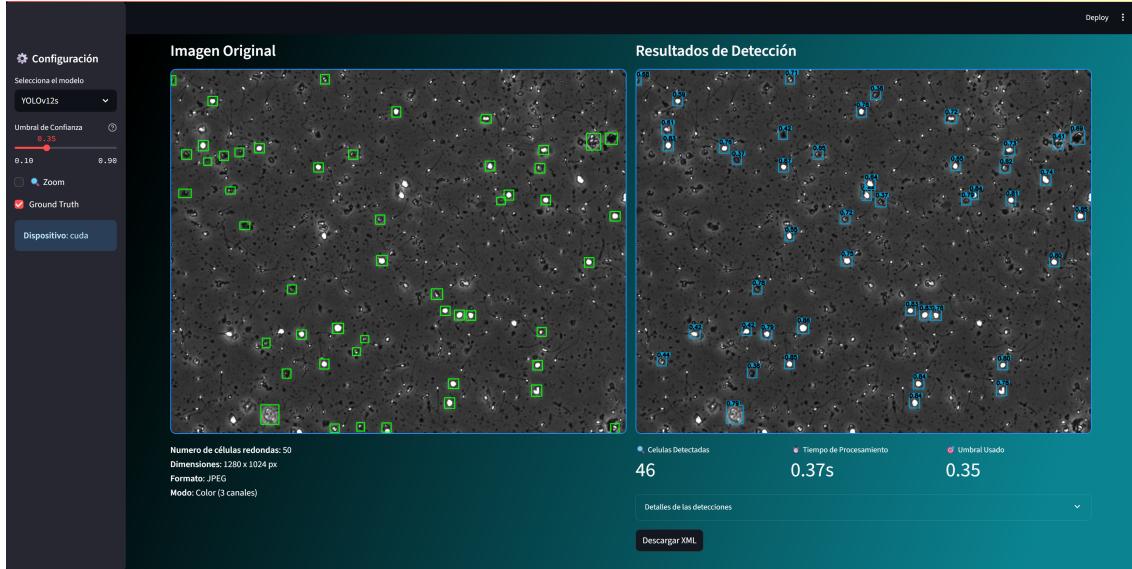


Figura C.2: Herramienta web: *checkbox ground truth*.

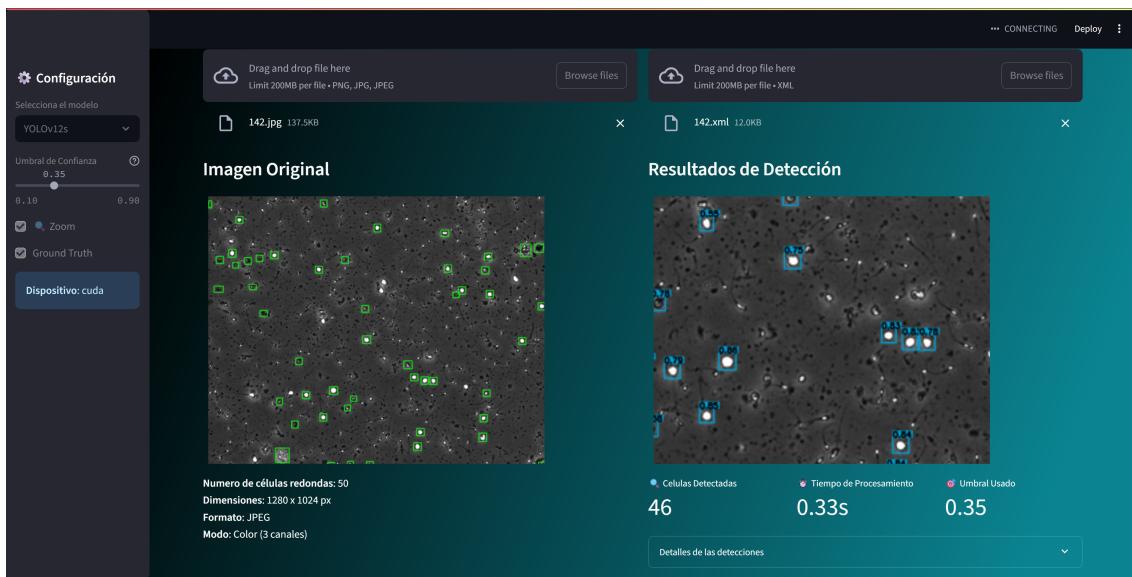


Figura C.3: Herramienta web: *checkbox zoom*.

Cabe destacar que el *widget* "Descargar XML" permite al usuario descargar las predicciones del modelo realizadas sobre la imagen de entrada en un formato PASCAL VOC.

Si el usuario prefiere evaluar un conjunto de imágenes con sus respectivas anotaciones, se debe seleccionar la elección de "carpeta completa" como se muestra en la Figura C.4.

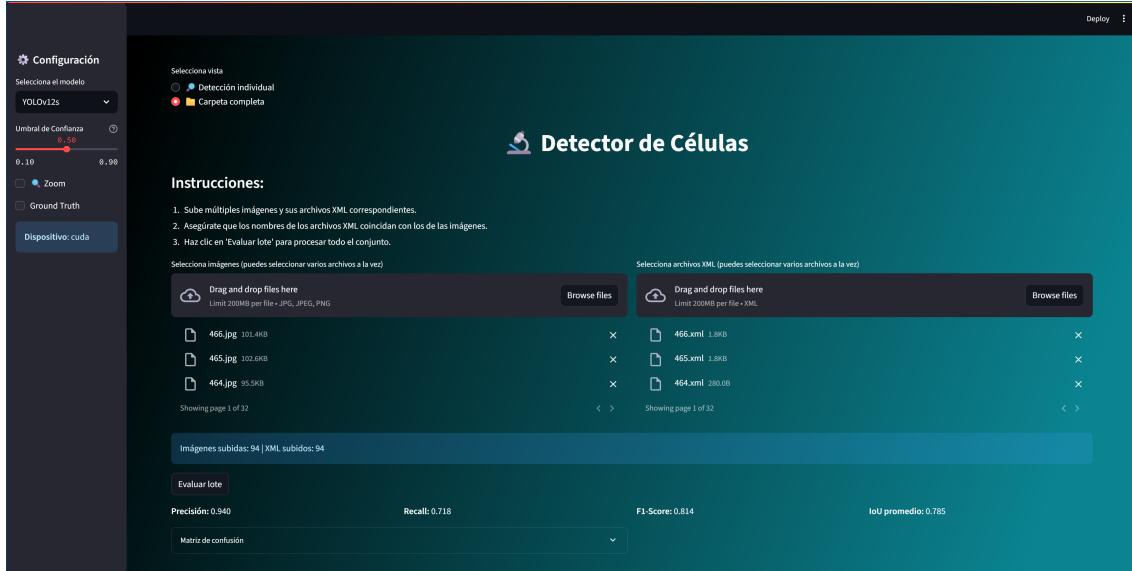


Figura C.4: Herramienta web para procesar un conjunto de imágenes.

En este modo de evaluación, no está permitido el uso de los *widgets*: *ground truth* y *zoom*. Sin embargo, el procesamiento del modelo seleccionado junto con su correspondiente umbral de confianza para el IoU, está acompañado de un pequeño conjunto de métricas: *precision*, *recall*, *F1-score* y *mAP*. Para una mejor evaluación, la herramienta muestra una matriz de confusión (Figura C.5).

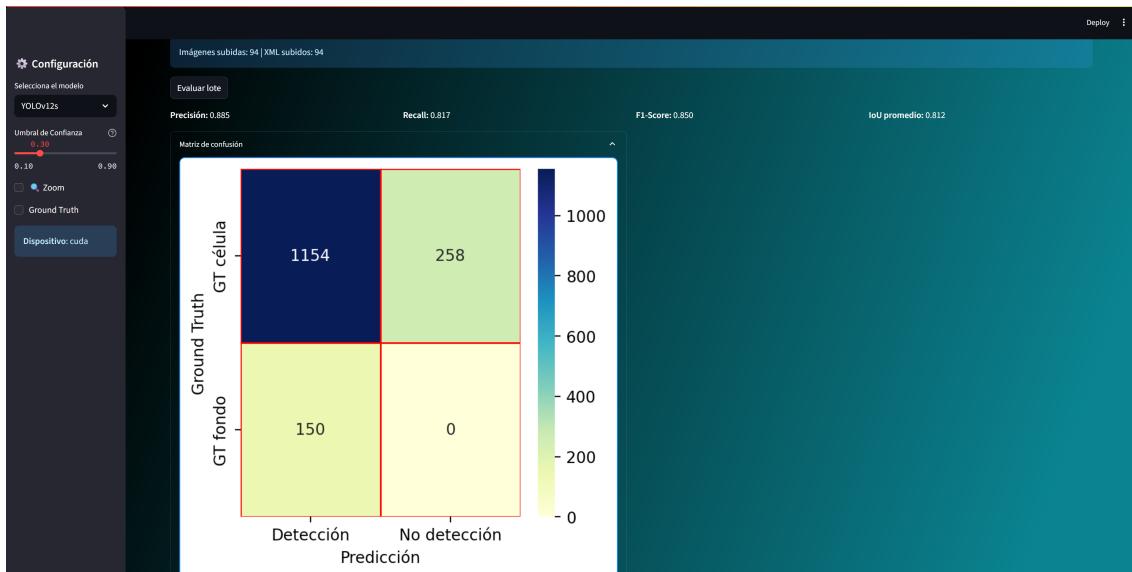


Figura C.5: Herramienta web: métricas y matriz de confusión.

C.2. Recomendaciones de uso

A continuación, se recogen las pautas prácticas para el correcto funcionamiento de la herramienta web.

- **Entorno recomendado:** se recomienda usar un entorno conda para tener aisladas las dependencias del proyecto y evitar así, conflictos entre dependencias. Desde el directorio 04.Code se proponen dos alternativas:
 - Instalación del entorno completo con conda (recomendado):
 - `conda env create -f environment.yml`
 - `conda activate tfm_env`
 - Instalación de un subconjunto de dependencias con conda:
 - `conda create -n tfm_env python=3.11 -y`
 - `conda activate tfm_env`
 - `pip install -r 04.Codigo/requirements.txt`
- **Ejecución de la aplicación:** en la carpeta 04.Codigo/cell_detection_App ejecutar:
 - `streamlit run app.py`
 - Automáticamente se abre en el navegador por defecto la URL que *Streamlit* indique (por defecto `http://localhost:8501`).
- **Compatibilidad con modelos:** limitada a modelos de YOLO [5].

C.3. Resolución de problemas

Problemas que pueden tener lugar durante la instalación:

- **La app no arranca o *Streamlit* muestra error:**
 - Verificar dependencias: `pip install -r 04.Codigo/requirements.txt`
 - Comprobar la salida en el terminal donde se ejecuta `streamlit run app.py` y revisar trazas de error.
 - Borrar caché de *Streamlit*: `streamlit cache clear`.

■ Modelo no encontrado o error de carga de pesos:

- Confirmar que el fichero de pesos existe en `models/` y que la ruta es correcta (`04.Codigo/cell_detection_App/models/`).

■ Problemas con GPU o CUDA:

- Comprobar estado de la GPU: ejecutar `nvidia-smi` en terminal.
- Verificar que PyTorch detecta la GPU:
 - `python -c "import torch; print(torch.cuda.is_available())"`
- Si falla, forzar ejecución en CPU desde la UI o variable de entorno: `CUDA_VISIBLE_DEVICES=""`.

■ Errores al procesar imágenes o XML mal formados:

- Validar que las imágenes están en formatos soportados (`.jpg`, `.jpeg`, `.png`) y que los XML cumplen con el formato PASCAL VOC.
- Revisar el log de parsing en `04.Codigo/cell_detection_App/utils/` y corregir los ficheros señalados.

Anexo D

Evaluación comparativa de arquitecturas

En este anexo, se recojen las representaciones gráficas para la evaluación de los modelos sobre los diferentes conjuntos de *test*. Todas las representaciones en las que no se hace notar el IoU de confianza del modelo, tienen por defecto un IoU de 0,5. Las representaciones a evaluar recogen información a través de matrices de confusión, curva F1-confianza y resultados cualitativos del modelo en tiempo real.

D.1. Matrices de confusión

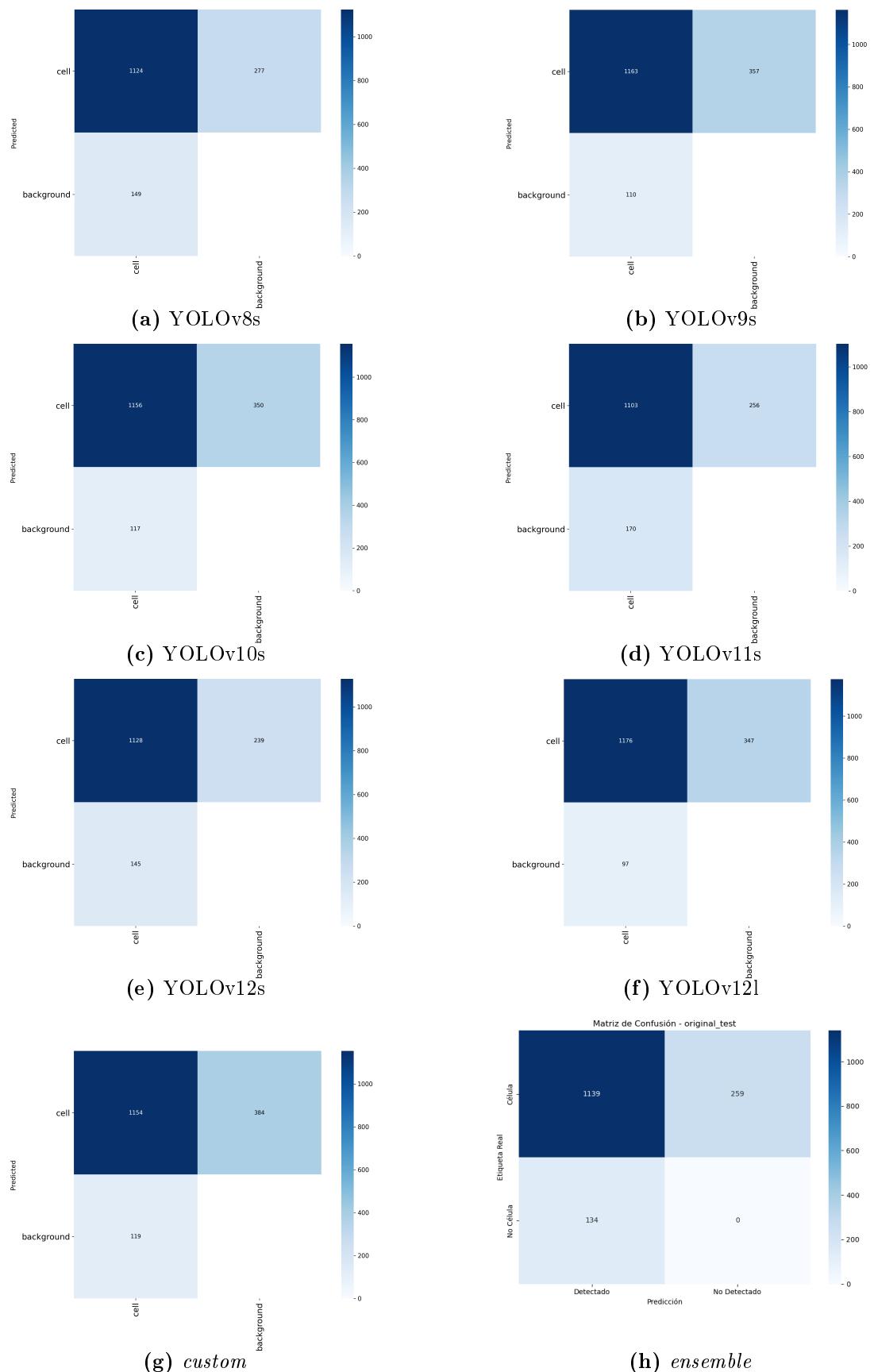
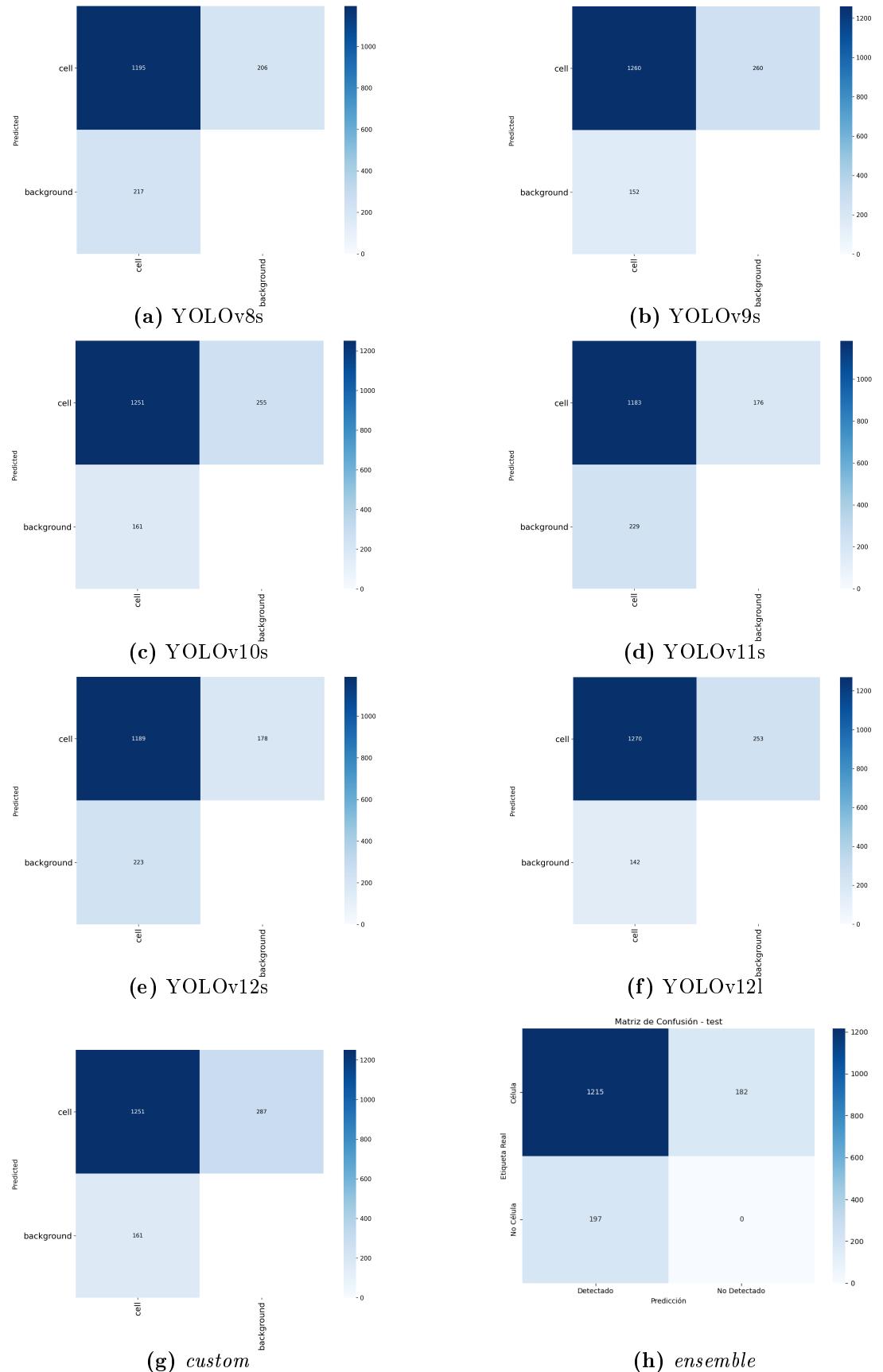
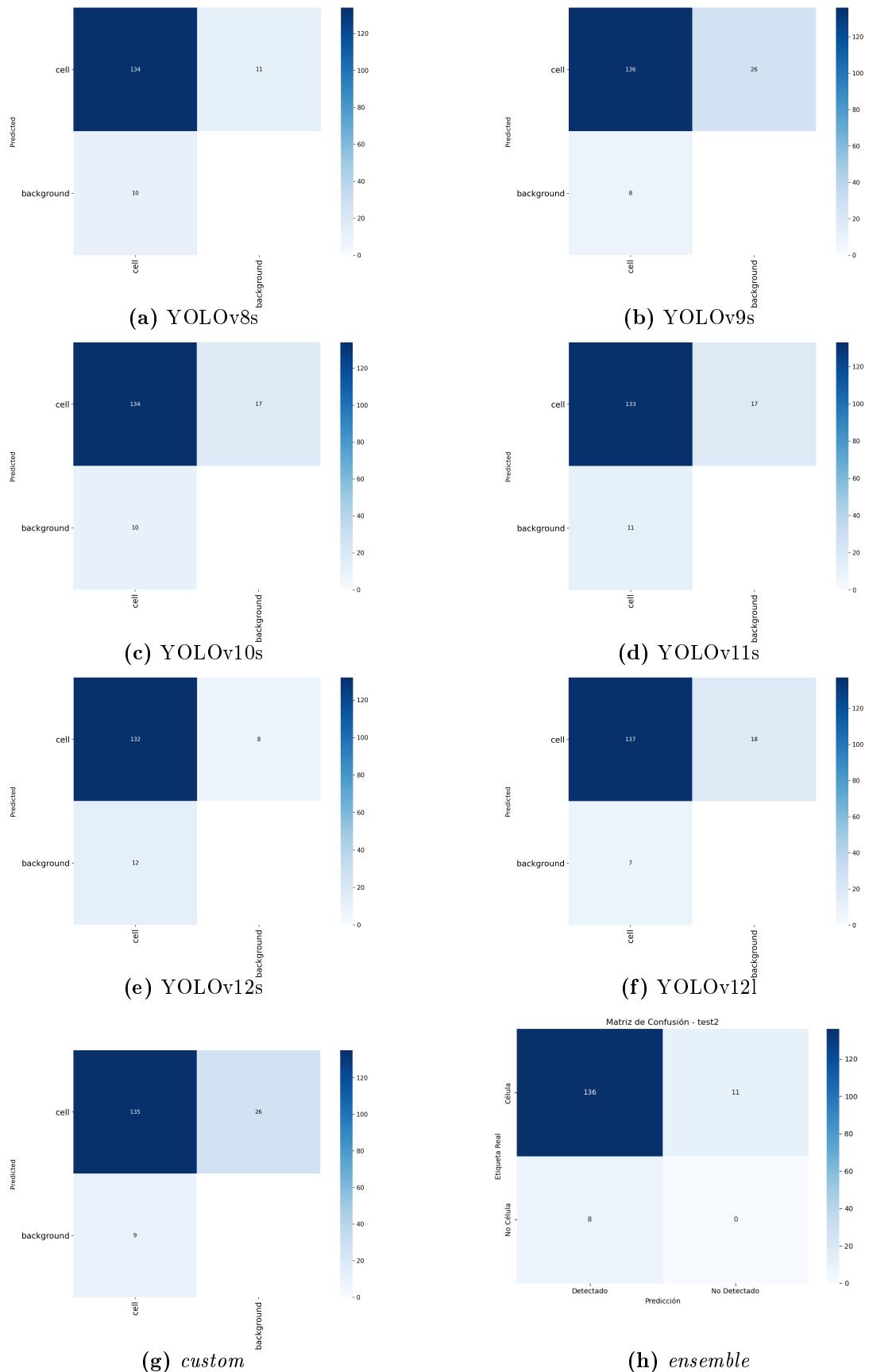
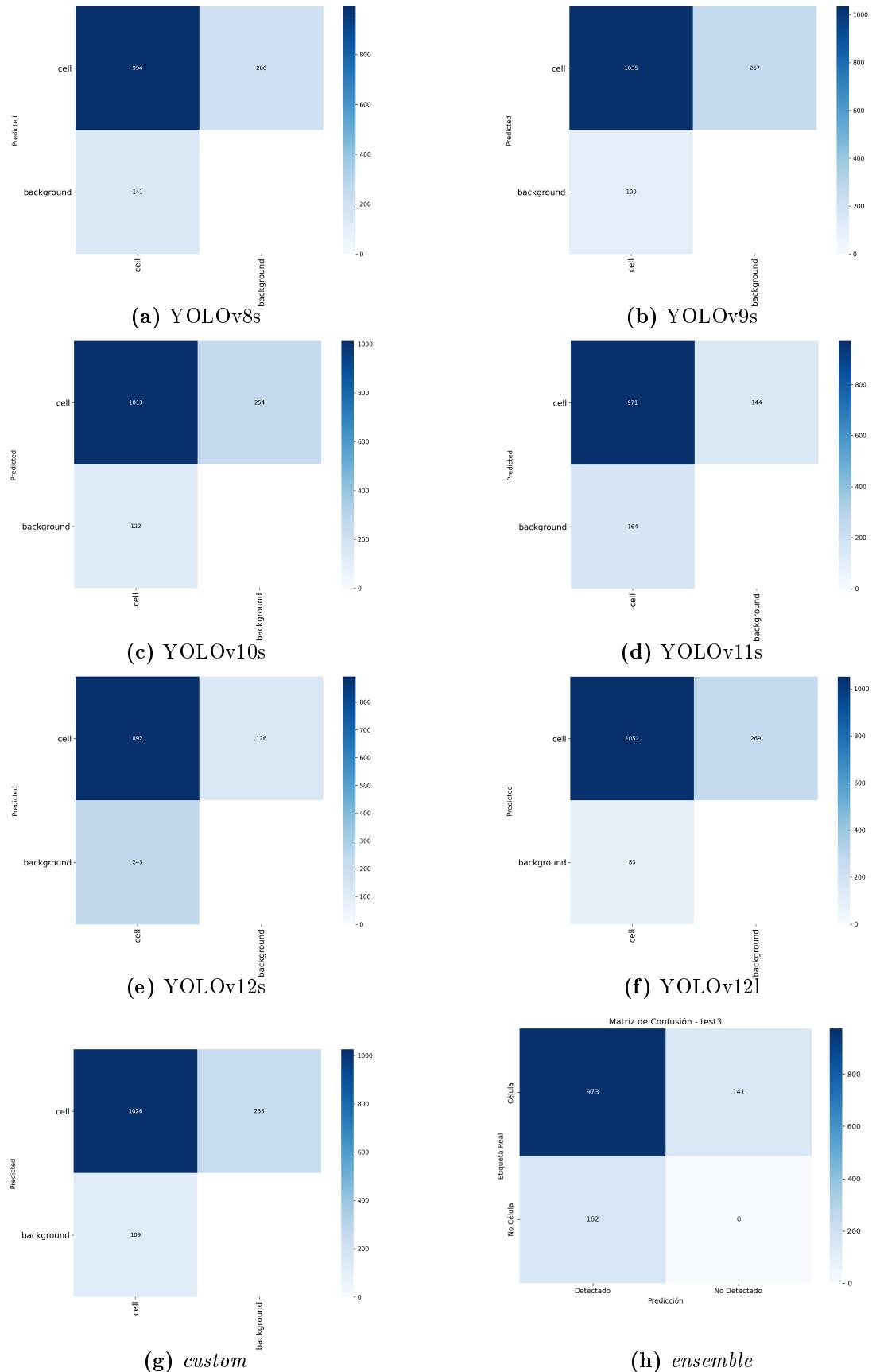


Figura D.1: Matrices de confusión para los modelos evaluados en original_test

**Figura D.2:** Matrices de confusión para los modelos evaluados en test

**Figura D.3:** Matrices de confusión para los modelos evaluados en test2

**Figura D.4:** Matrices de confusión para los modelos evaluados en test3

D.2. Curvas F1-confianza

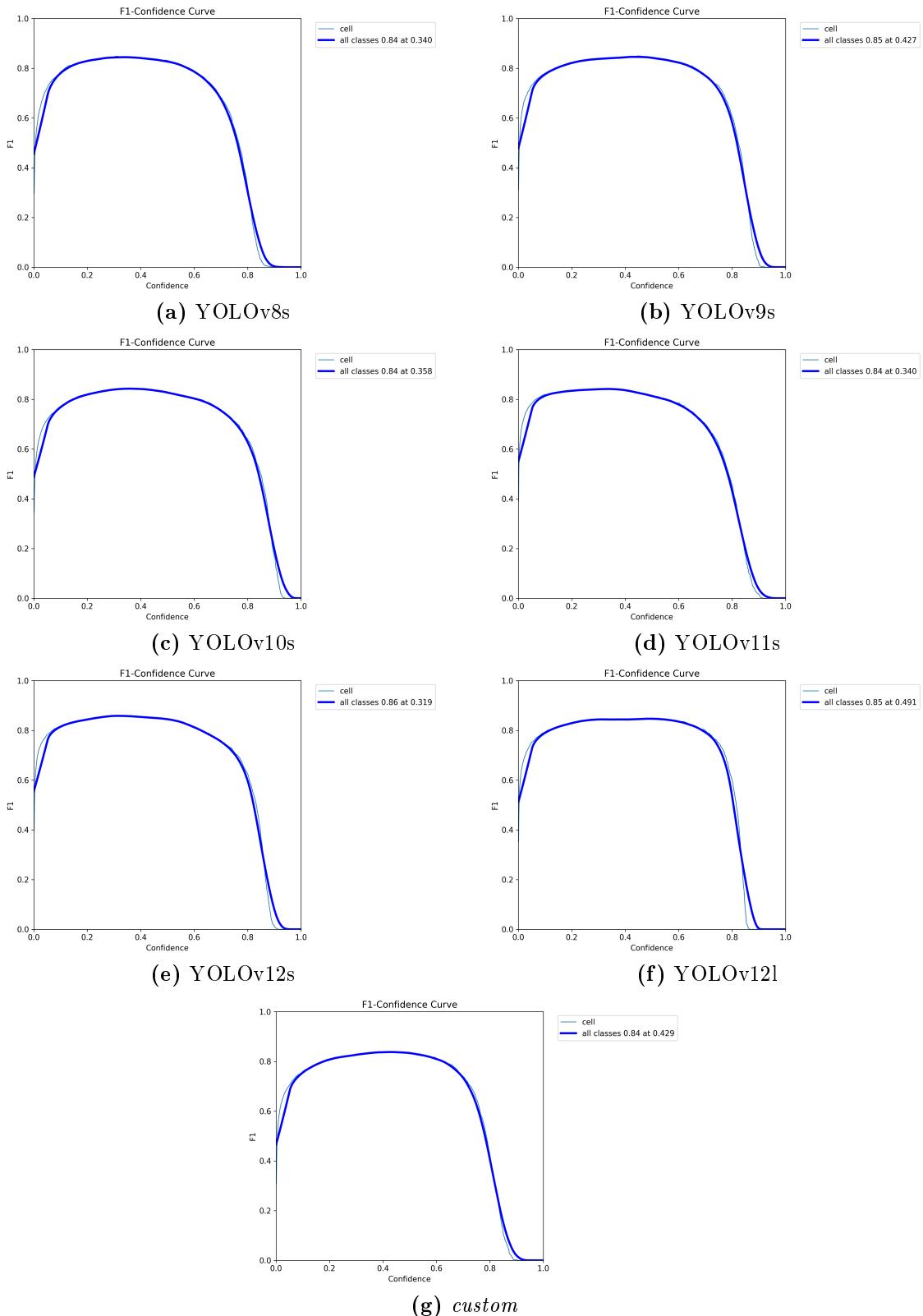
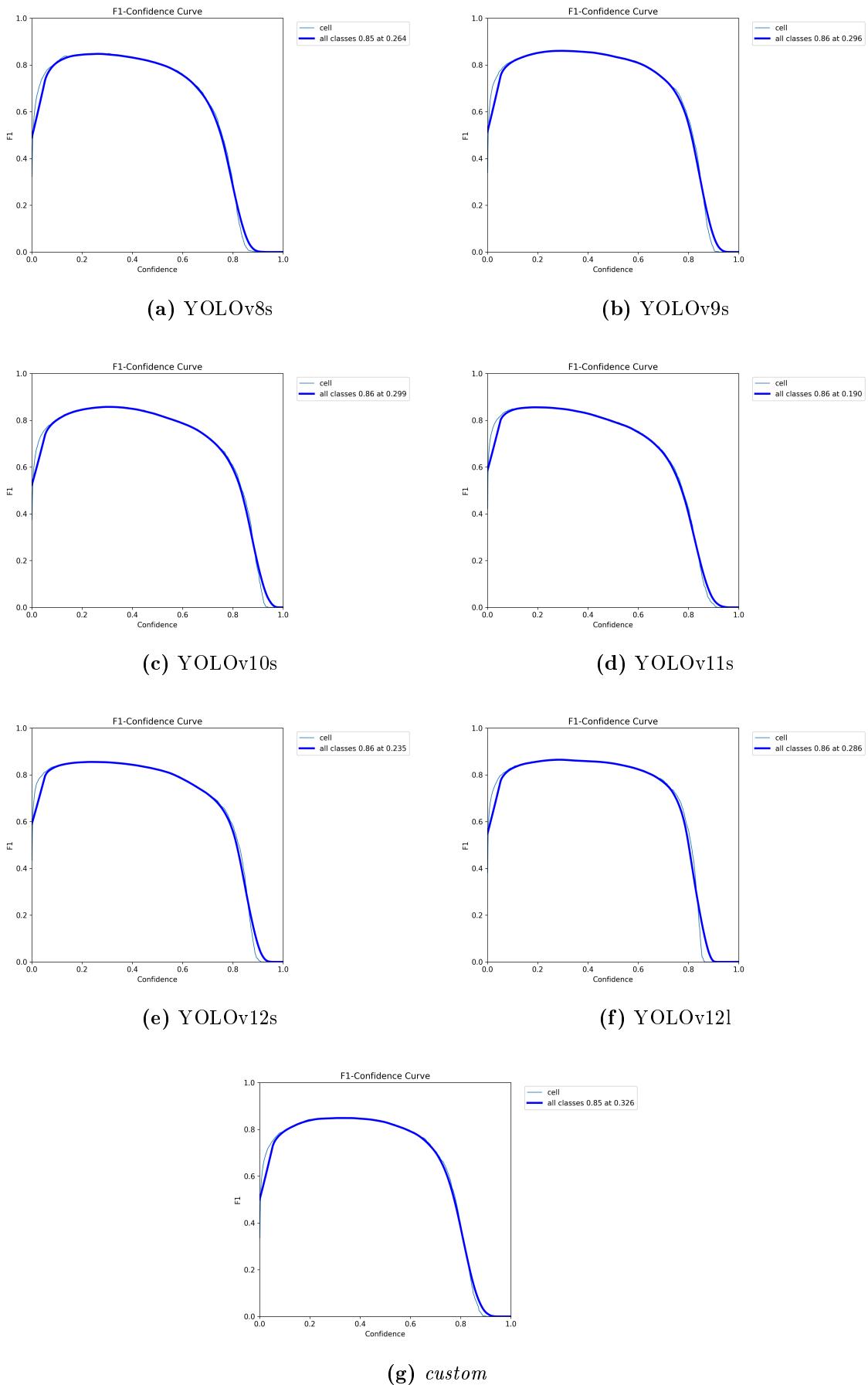


Figura D.5: Curvas F1-confianza para los modelos evaluados en original_test

**Figura D.6:** Curvas F1-confianza para los modelos evaluados en test

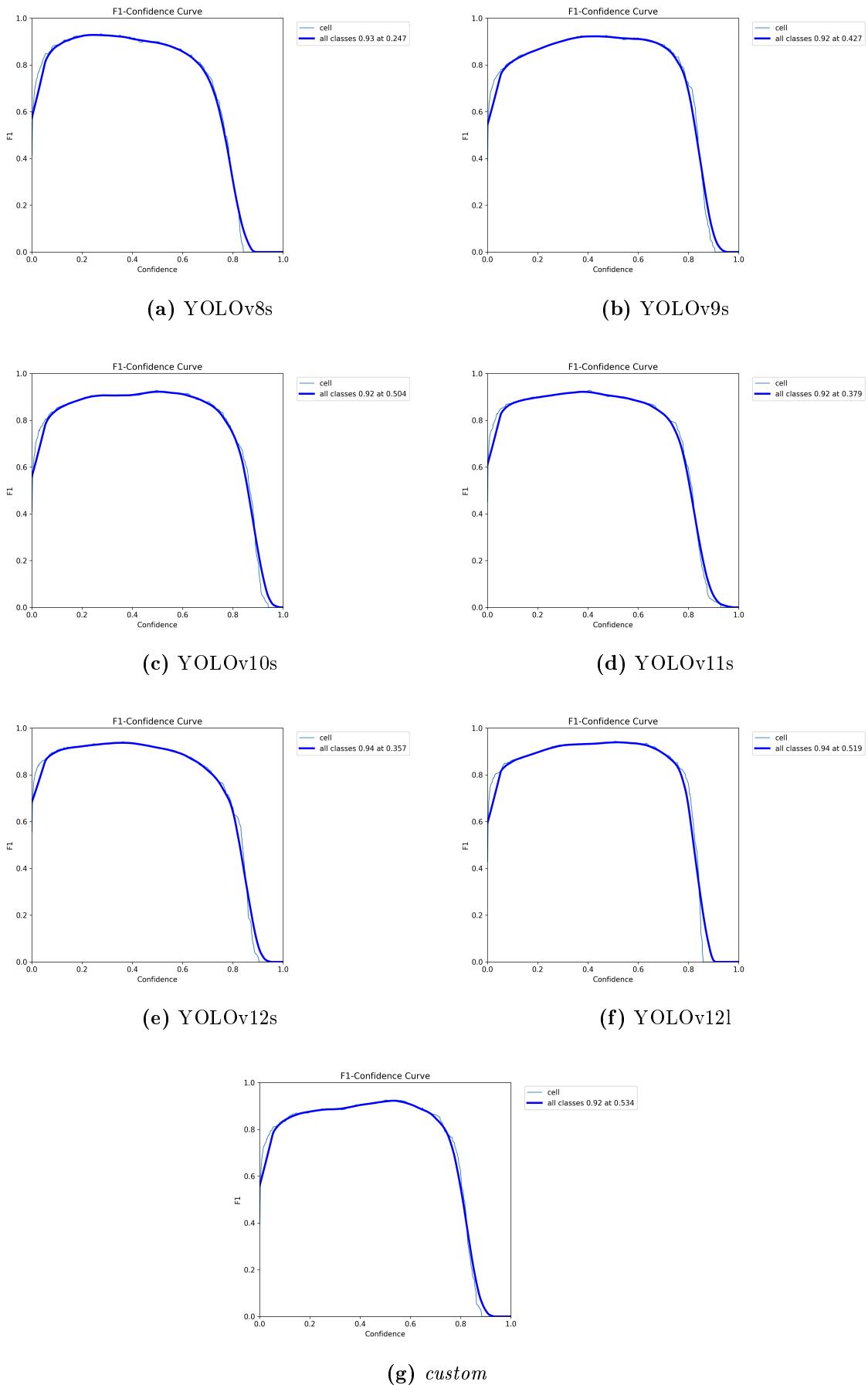
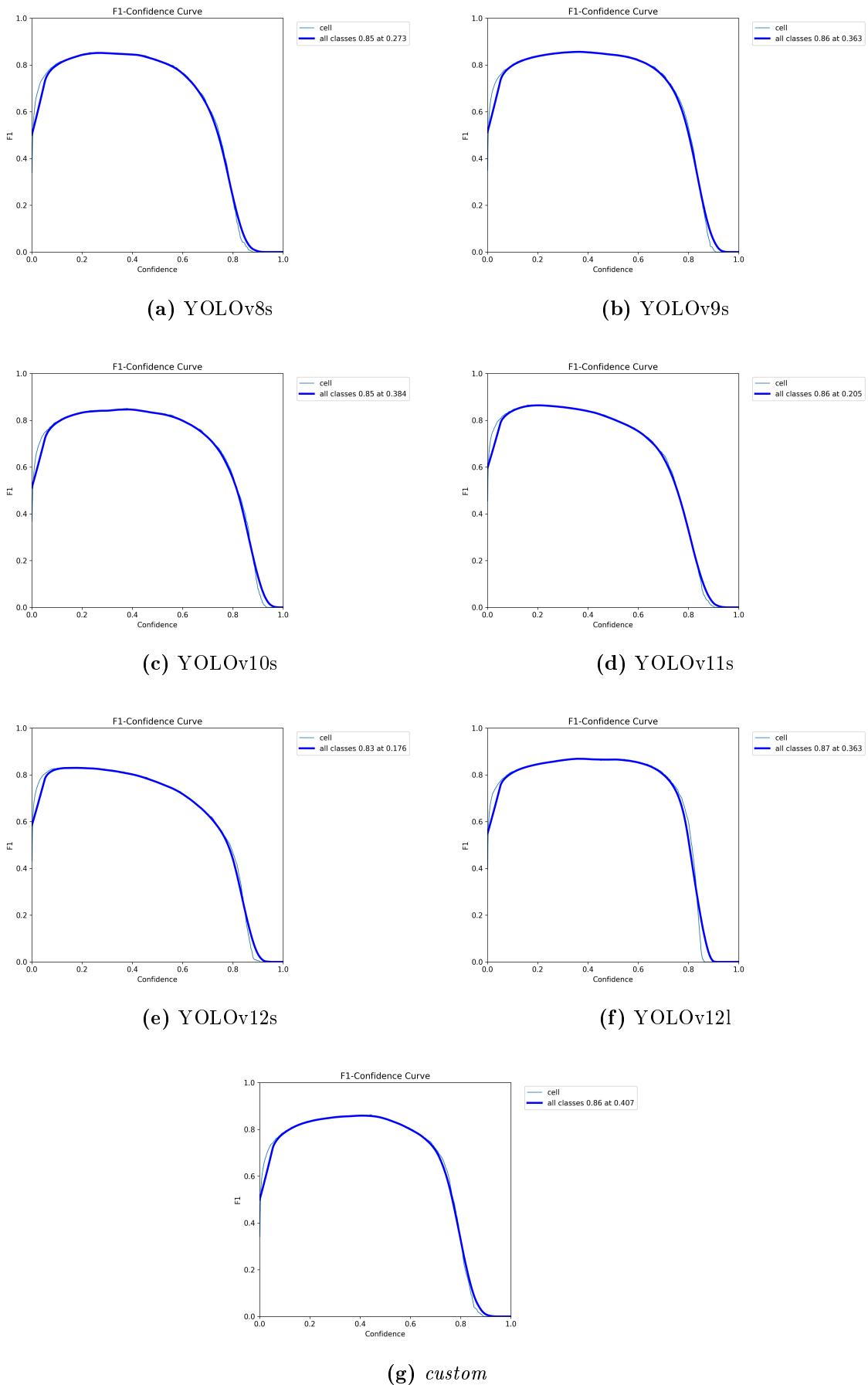
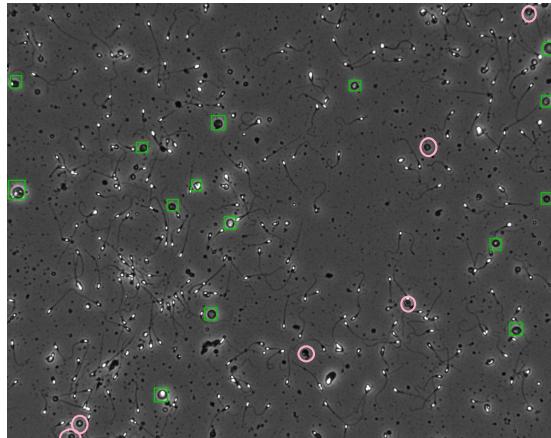


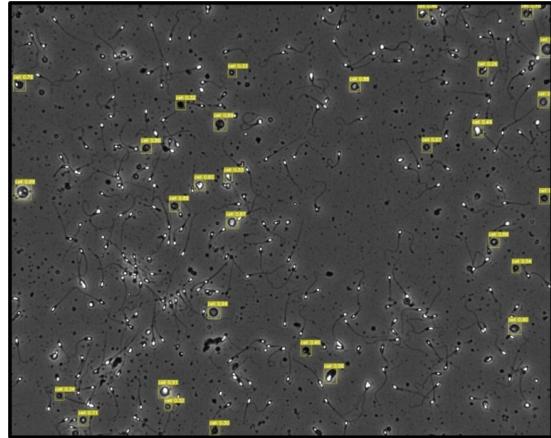
Figura D.7: Curvas F1-confianza para los modelos evaluados en test2

**Figura D.8:** Curvas F1-confianza para los modelos evaluados en test3

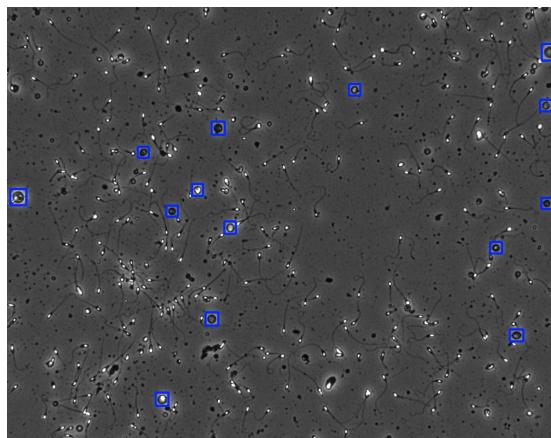
D.3. Resultados cualitativos



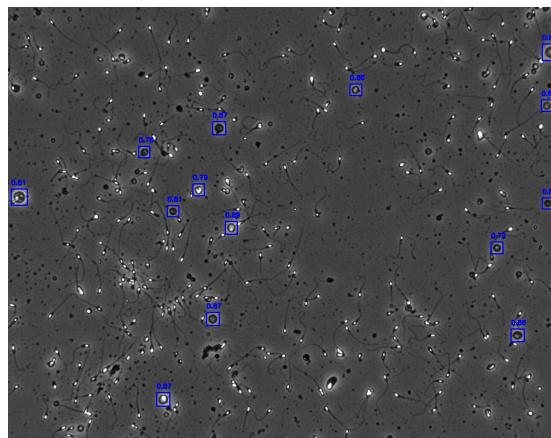
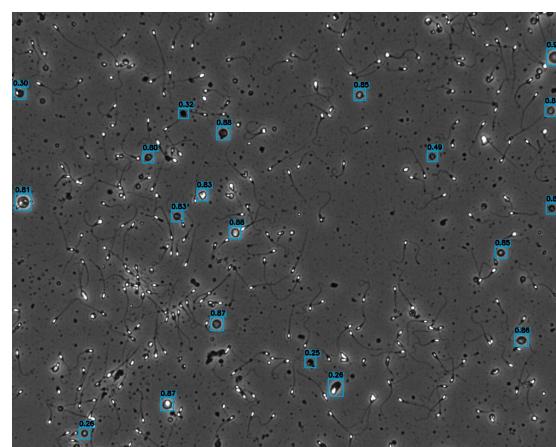
(a) Imagen evaluada por experto



(b) Prueba de concepto inicial



(c) Modelo YOLOv12s con IoU 0.5

(d) Modelo *ensemble*

(e) Modelo YOLOv12s con IoU 0.25

Figura D.9: Resultados comparativos para la Figura 7

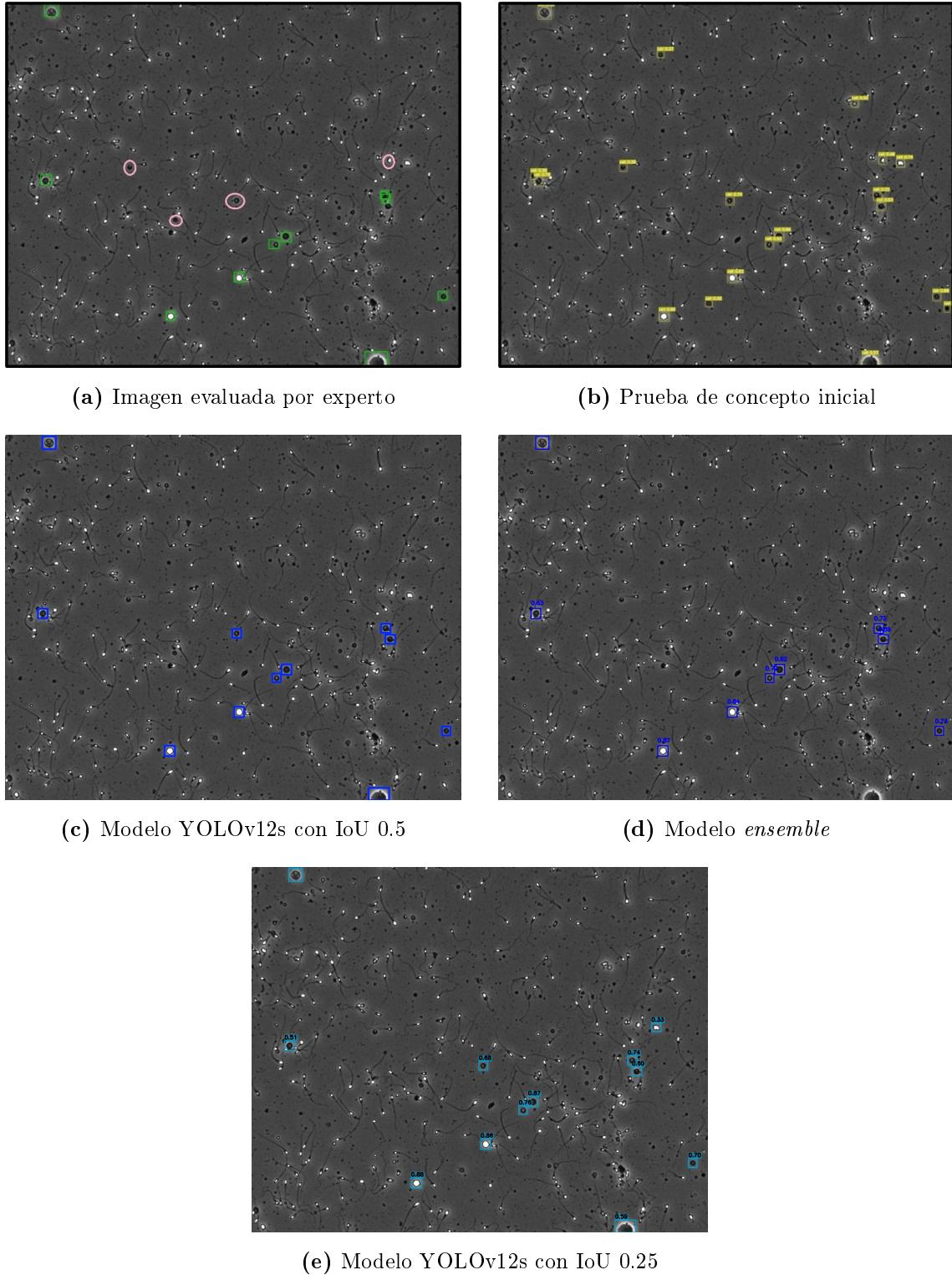


Figura D.10: Resultados comparativos para la Figura 9

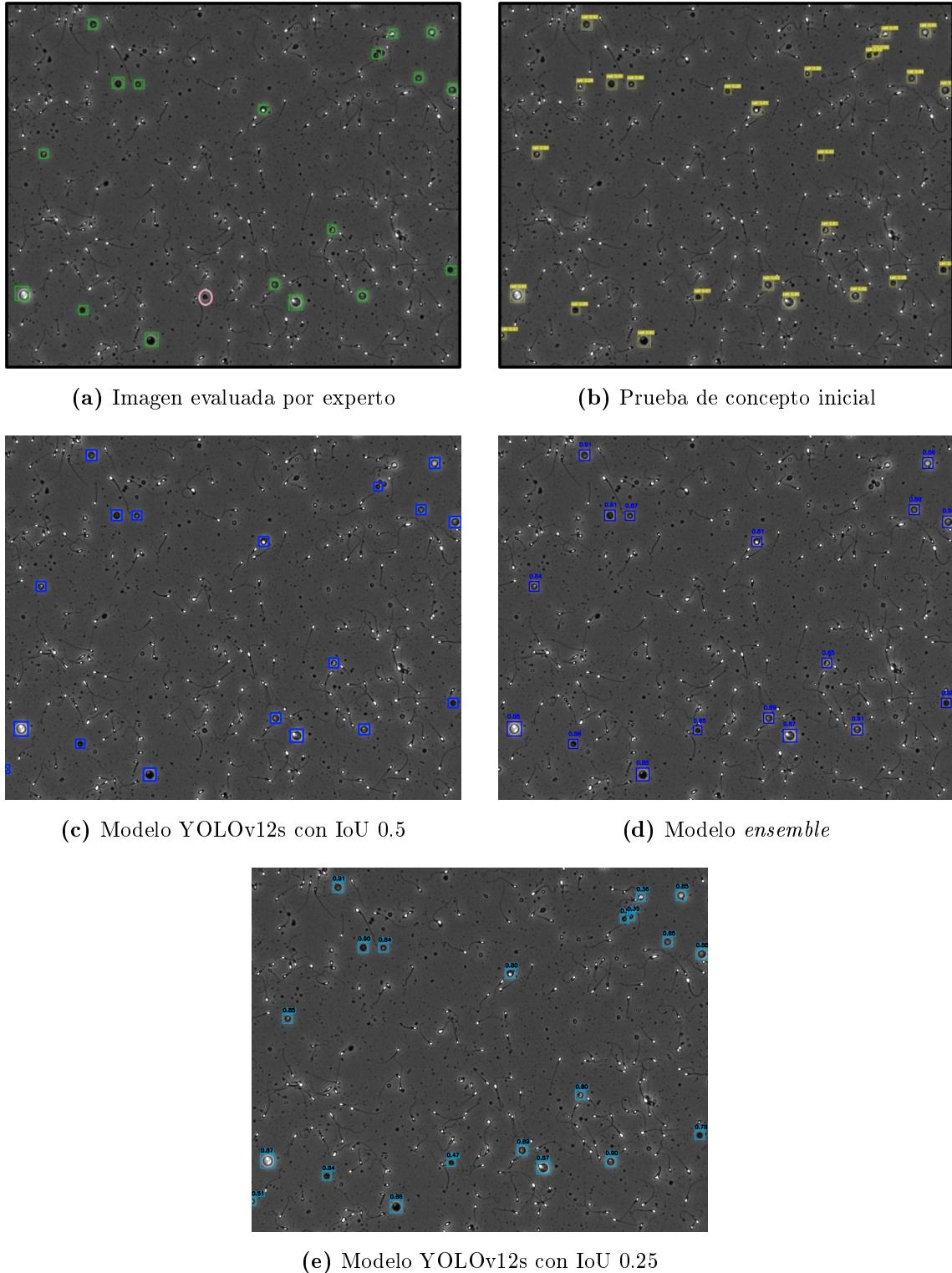


Figura D.11: Resultados comparativos para la Figura 38

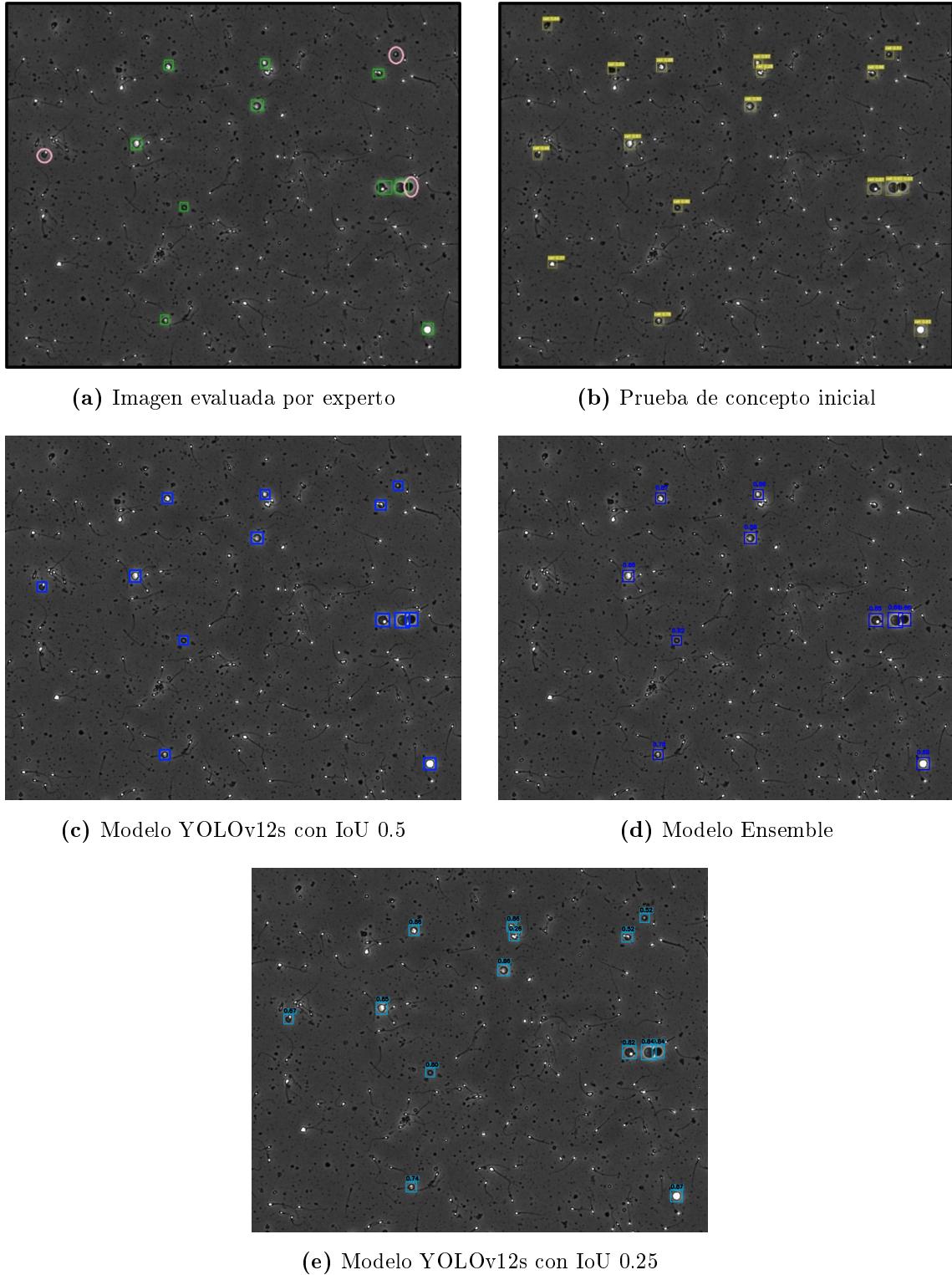


Figura D.12: Resultados comparativos para la Figura 95

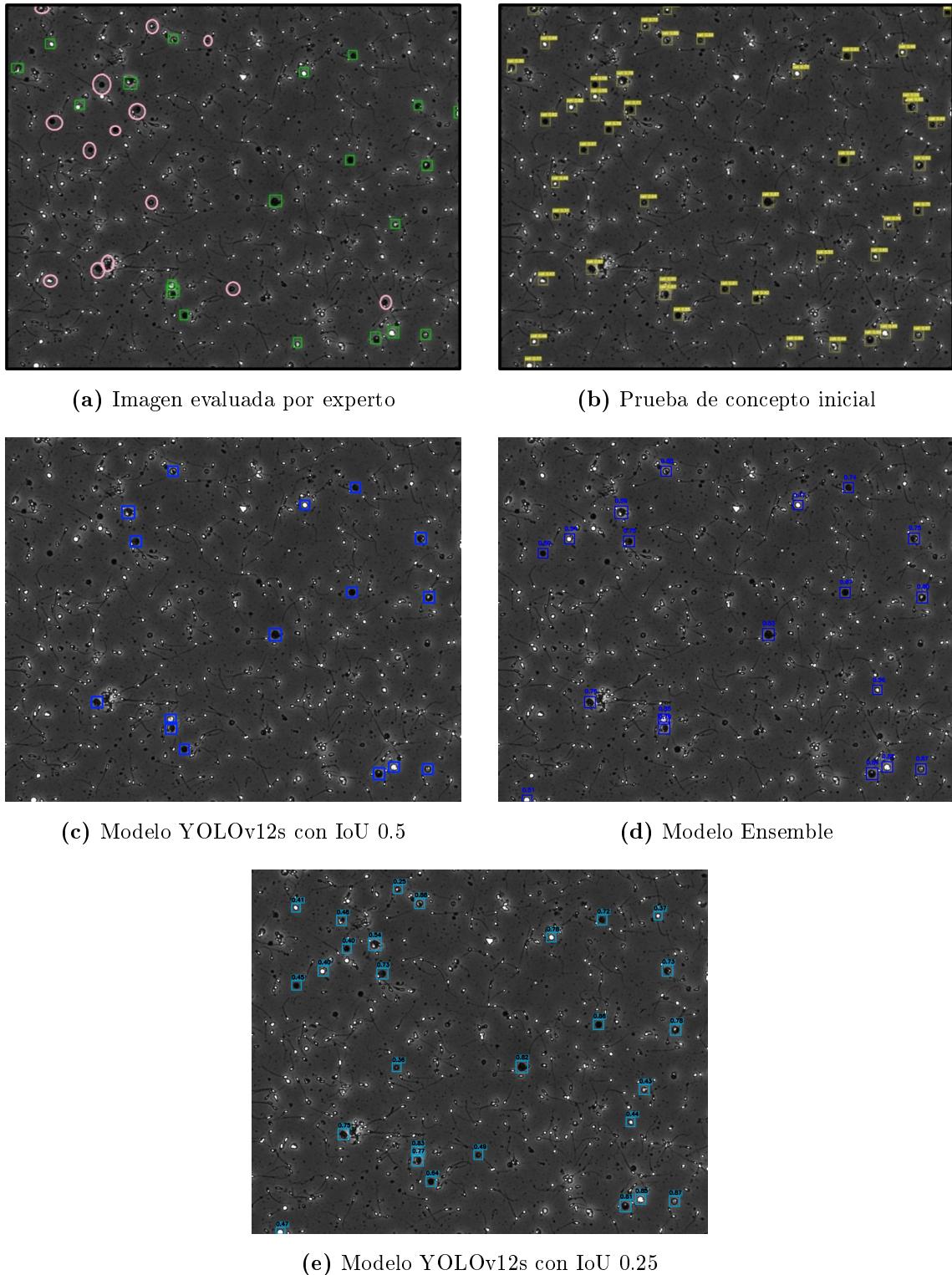


Figura D.13: Resultados comparativos para la Figura 139

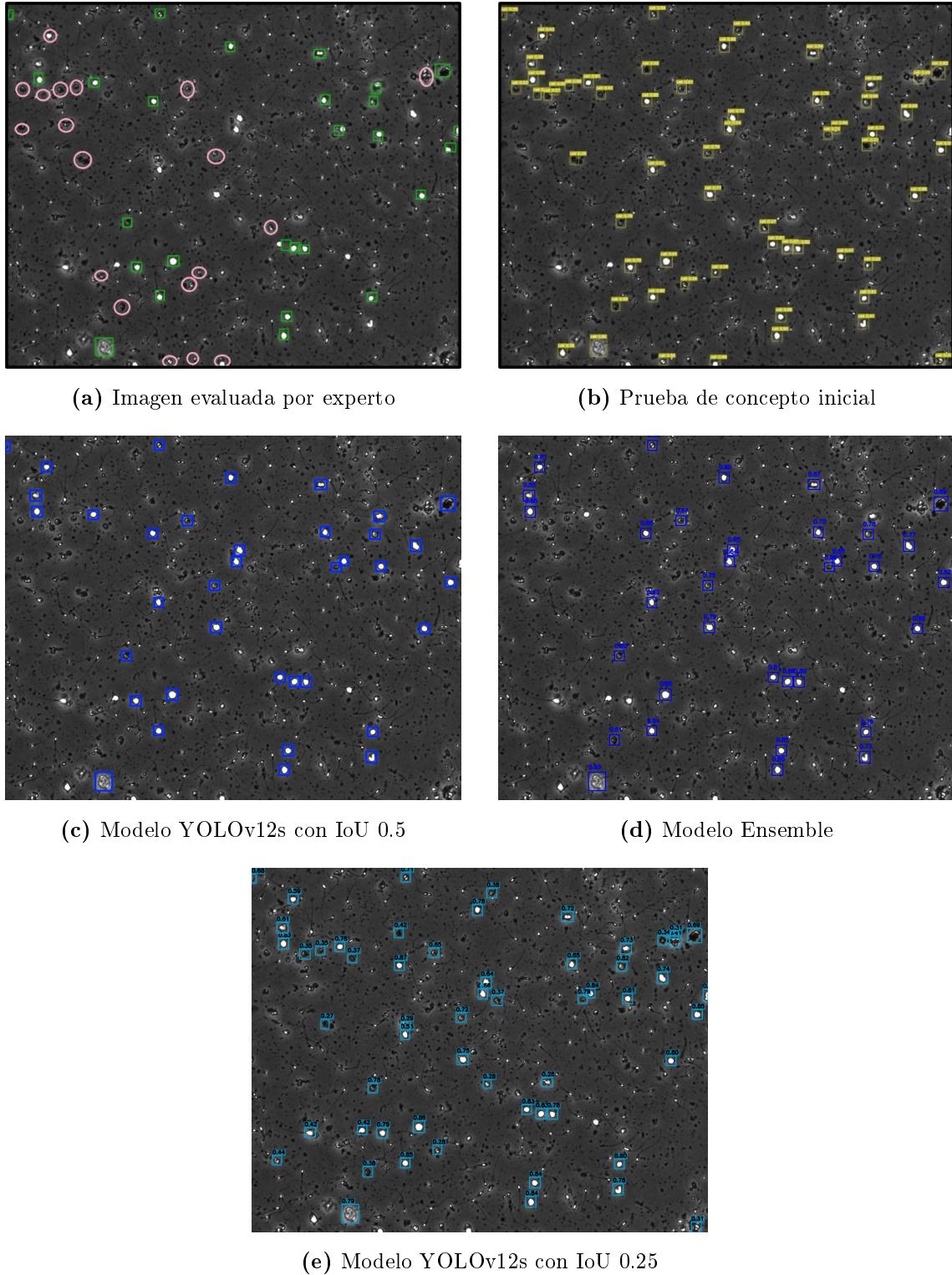


Figura D.14: Resultados comparativos para la Figura 142

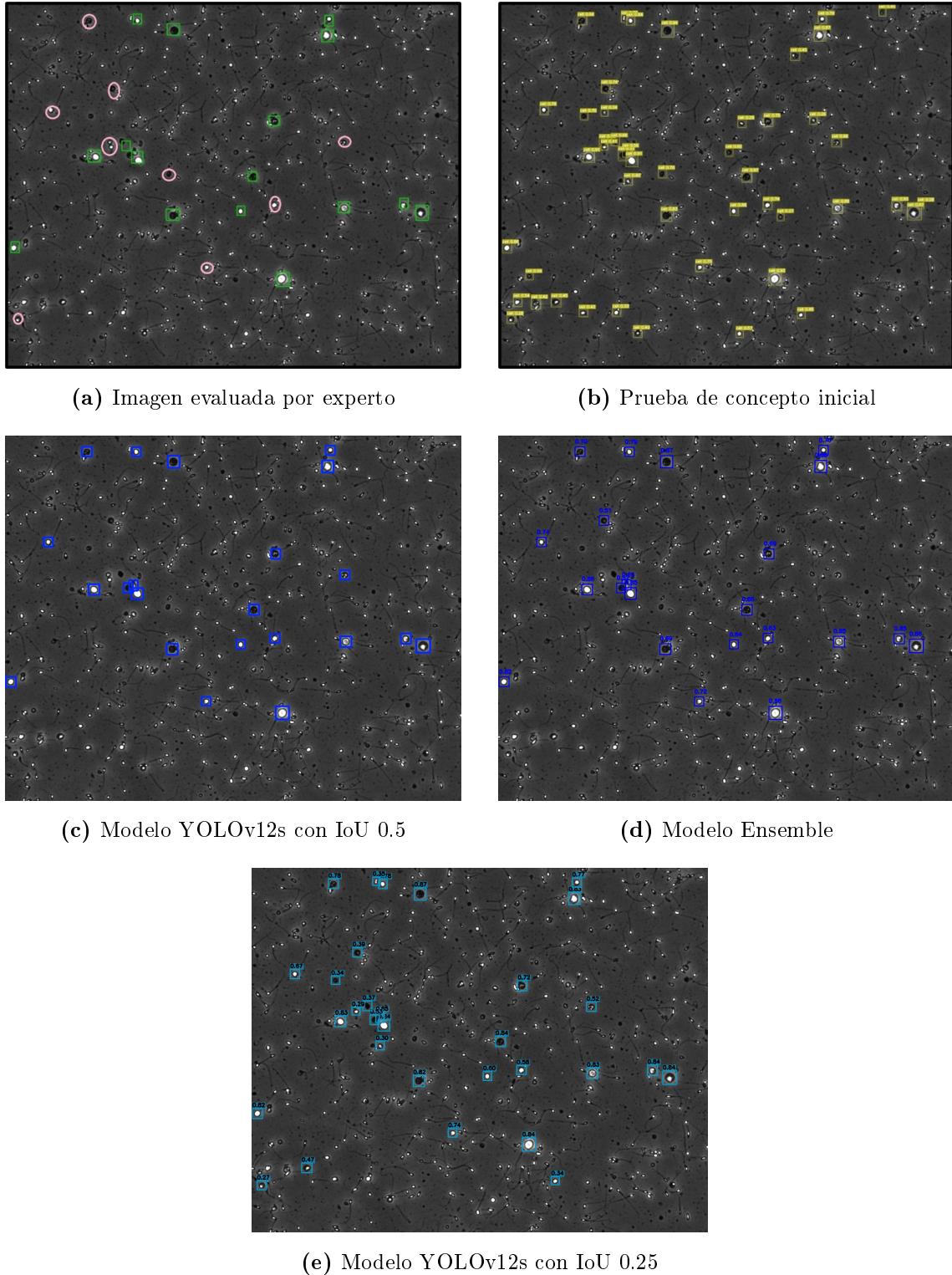


Figura D.15: Resultados comparativos para la Figura 221

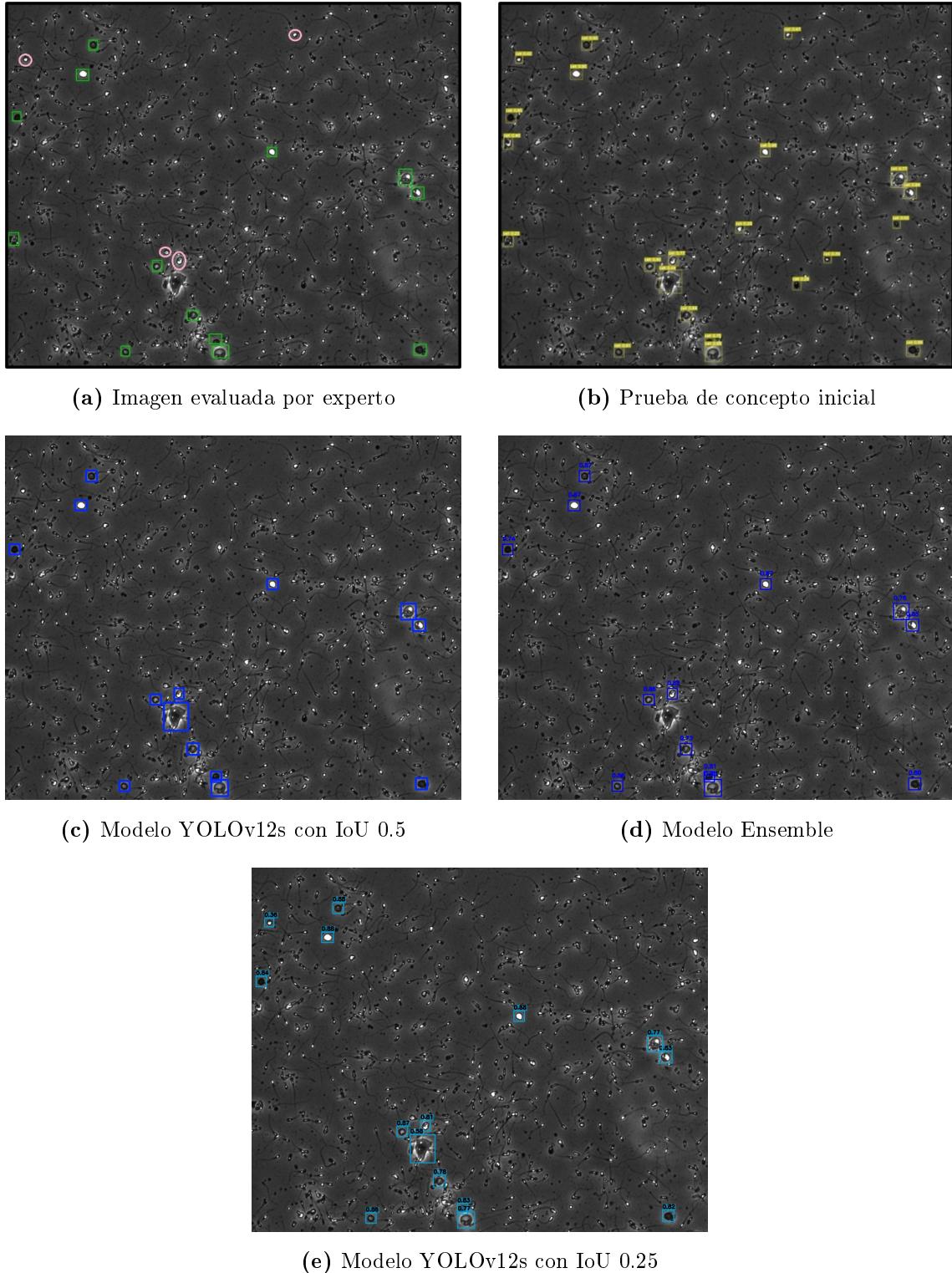


Figura D.16: Resultados comparativos para la Figura 347

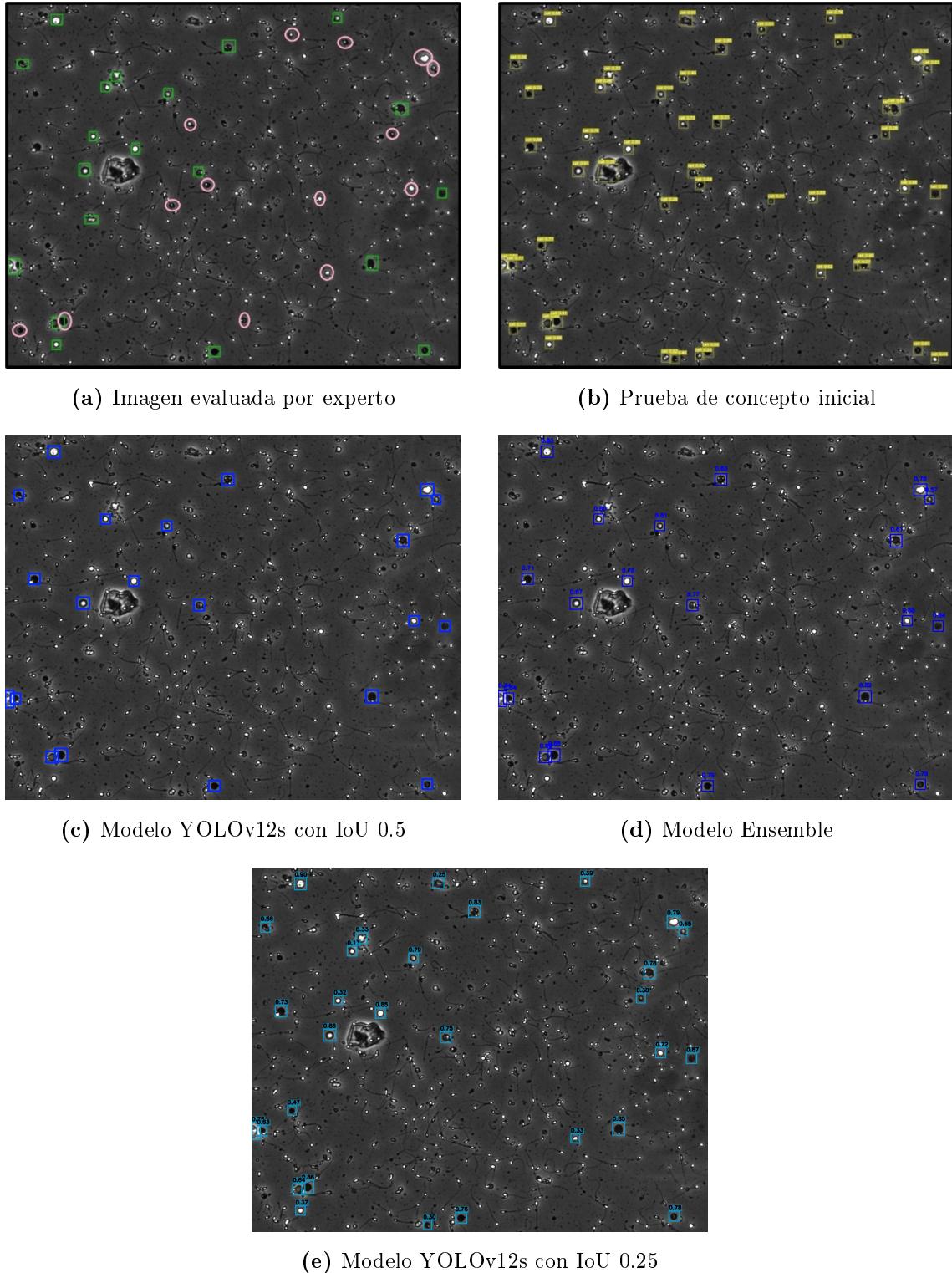


Figura D.17: Resultados comparativos para la Figura 415

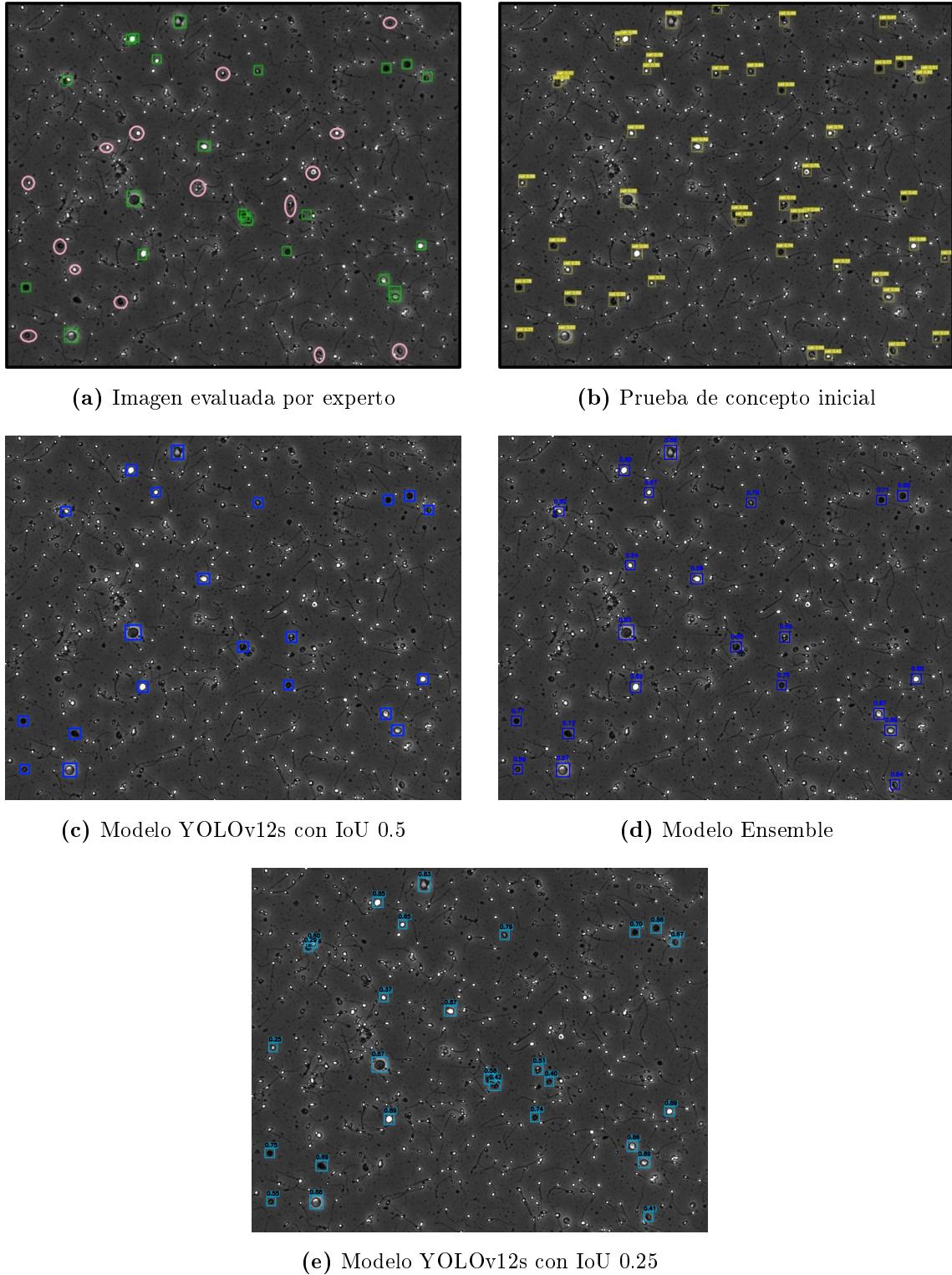


Figura D.18: Resultados comparativos para la Figura 461