TecnoLógicas

Inspección visual de defectos en placas de circuito impreso utilizando arquitecturas YOLO: Una comparación sobre el conjunto de datos DeepPCB

Visual Defect Detection in Printed Circuit Boards using YOLO Architectures: A Comparison on the DeepPCB Dataset

Alexia Blanco[1]; Ivon Oristela Benítez-González[2,]; José Manuel Bernal-de Lázaro[2,3]

[1]Maestría en Robótica y Automatización, Universidad Internacional de Valencia (VIU), Valencia - España
[2]Escuela Superior de Ingeniería y Tecnología, Universidad Internacional de La Rioja (UNIR), La Rioja – España
[3]Instituto Politécnico-Universidade do Estado do Rio de Janeiro (IPRJ-UERJ), Rio de Janeiro - Brasil

Correspondencia: jose.lazaro@iprj.uerj.br

Recibido: 22 Septiembre 2025 **Aceptado:** 24 Julio 2026 **Disponible:** 28 Julio 2026

Cómo citar / How to cite
A. Blanco, I. O. Benítez-González, and J. M. Bernal-de Lázaro, "Inspección visual de defectos en placas de circuito impreso utilizando arquitecturas YOLO: Una comparación sobre el conjunto de datos DeepPCB" *TecnoLógicas*, vol. 29, no. 66, e3665, 2026. https://doi.org/10.22430/22565337.3665

CC BY NC SA

Resumen

La detección de defectos en placas de circuito impreso (PCB, por sus siglas en inglés) continúa siendo un desafío para los sistemas modernos de manufactura electrónica, donde la necesidad de reducir costos y mejorar la confiabilidad de los productos ha impulsado el desarrollo y aplicación de nuevos modelos de inteligencia computacional. En particular, la familia de arquitecturas YOLO ha evolucionado a lo largo de múltiples versiones, consolidándose como una de las principales soluciones para la detección automática de defectos en sistemas de inspección visual en la industria electrónica. A partir de esto, el presente trabajo comparó YOLOv5 y YOLOv8 para la detección automática de fallos en placas de circuito impreso. La metodología empleada consistió en evaluar ambos modelos de visión computacional mediante el conjunto de imágenes DeepPCB, bajo un entorno de ejecución con soporte GPU, comparando su precisión, sensibilidad, F1-score y mAP con distintos umbrales de solapamiento. Asimismo, se evaluó cualitativamente la capacidad de generalización e inferencia de ambas arquitecturas utilizando imágenes no disponibles durante la etapa de entrenamiento. Los resultados alcanzados mostraron un elevado contraste entre ambas arquitecturas. YOLOv5m destacó por lograr el mejor equilibrio global entre precisión, sensibilidad y estabilidad, mientras que YOLOv8s presentó una mejor capacidad de localización de defectos bajo distintos umbrales. Finalmente, se concluye que YOLOv5 continúa siendo una alternativa adecuada para los procesos tradicionales de inspección visual en la industria electrónica, al ofrecer mayor eficiencia y escalabilidad en tareas de control de calidad, mientras que YOLOv8 representa una opción competitiva cuando se prioriza una mayor precisión en la localización de defectos.

Palabras clave

Detección de defectos, placas de circuito impreso, repositorio DeepPCB, visión artificial, YOLO.

Abstract

The detection of defects in Printed Circuit Boards (PCBs) remains a significant challenge in modern electronic manufacturing systems, where the need to reduce costs and improve product reliability has stimulated the development and application of new computational intelligence models. In this context, the YOLO architecture-based family has evolved through multiple versions, establishing itself as one of the leading solutions for automatic defect detection in visual inspection systems within the electronics industry. Based on this fact, the present work compared the YOLOv5 and YOLOv8 architectures for tasks focused on automatic fault detection in printed circuit boards. The methodology employed consisted of evaluating the computer vision models YOLOv5 and YOLOv8 using the DeepPCB image dataset in a GPU-enabled execution environment and comparing their precision, recall, F1-score, and mAP across different IoU thresholds. In addition, a qualitative assessment was conducted to evaluate the generalization and inference capabilities of both architectures, using images not included during the training stage. The results obtained revealed a clear contrast between the two architectures. YOLOv5m achieved the best overall balance between precision, recall, and detection stability, while YOLOv8s demonstrated superior defect localization capability under varying IoU thresholds. Overall, both computer vision models exhibited competitive performance in the DeepPCB dataset. Finally, it is concluded that YOLOv5 remains a suitable alternative for traditional visual inspection processes focused on the electronics industry due to its efficiency and scalability in quality control tasks, while YOLOv8 represents a competitive option when higher precision is prioritized in defect localization.

Keywords

Defect detection, printed circuit boards, deepPCB repository, machine vision, YOLO.

1. INTRODUCCIÓN

La creciente demanda de productos de elevada calidad propicia que la industria moderna incorpore nuevas soluciones tecnológicas para alcanzar mayor eficiencia, precisión y trazabilidad [1]-[3]. Por su importancia, la industria electrónica enfocada en la fabricación de placas de circuito impreso (PBC por sus siglas en inglés), constituye uno de los sectores líderes en la innovación tecnológica. Las PBC fabricados en esta industria se encuentran en una amplia gama de dispositivos. La existencia de defectos en PBC, incluso a nivel microscópico, puede comprometer el la fiabilidad y funcionamiento de los dispositivos electrónicos, incurriendo en pérdidas en los procesos que involucran dichos dispositivos [3]-[5].

Desde 1936 cuando Paul Eisler creó la placa de circuito impreso, el diseño y fabricación de PCB ha tenido una evolución continua, adicionando capas, materiales más resistentes y técnicas más avanzadas para la fabricación y miniaturización de las placas. Esto ha conllevado indudablemente a incrementar la fiabilidad de las PCB actuales [5]. Sin embargo, el proceso de producción de PCB es altamente complejo e involucra múltiples etapas, que incluyen el diseño, grabado químico, perforación, aplicación de capas protectoras y montaje de componentes [6], [7]. En cada uno de estos procedimientos pueden surgir defectos de diferente índole [4], [5]. Algunos ejemplos típicos se muestran en la Figura 1, donde se evidencian cortes de pistas (open), cortocircuitos (short), perforaciones no deseadas (mousebite), cobre residual (copper), filamentos metálicos (spur) o imperfecciones en la soldadura (pin hole) [8]. La detección temprana de tales defectos o anomalías es vital para garantizar la continuidad de la cadena de fabricación, y cumplir con los estándares internacionales que regulan la industria electrónica.

Es extremadamente frecuente que la inspección de placas PCB se realice de manera manual o utilizando sistemas de visión artificial tradicionales basados en reglas. Sin embargo, la inspección manual presenta una elevada dificultad y es altamente sensible a la subjetividad humana. Un gran número de placas PCB pasan al mercado por los errores humanos al intentar identificar defectos de tamaño reducido y difícil localización, fundamentalmente en contextos con una elevada producción de placas PCB [9], [10]. El uso de sistemas inteligentes de detección de defectos basados en reglas permite superar esta limitación, pero su éxito depende en gran medida de las condiciones de iluminación y su configuración previa, lo que limita su robustez en escenarios que involucran ruido visual y/o la existencia de defectos con apariencia irregular. En este contexto, el uso de herramientas de visión computacional que utilizan aprendizaje profundo se ha convertido en una opción muy atractiva [10], [11], proporcionando una gran

diversidad de herramientas que consiguen aprender directamente de datos y adaptarse a la variabilidad típica de la industria electrónica. Bajo este enfoque, son varias las arquitecturas neuronales basadas en aprendizaje profundo que han sido reportadas y aplicadas para la inspección de defectos en placas PCB, entre ellas Faster R-CNN, DECNet, ResNet, U-Net y la familia YOLO [12]-[18].Las arquitecturas basadas en la familia YOLO se han convertido particularmente en una de las más populares por su equilibrio entre precisión y velocidad durante la detección de objetos en tiempo real [11], [17], lo cual es crítico en líneas de producción automatizadas.

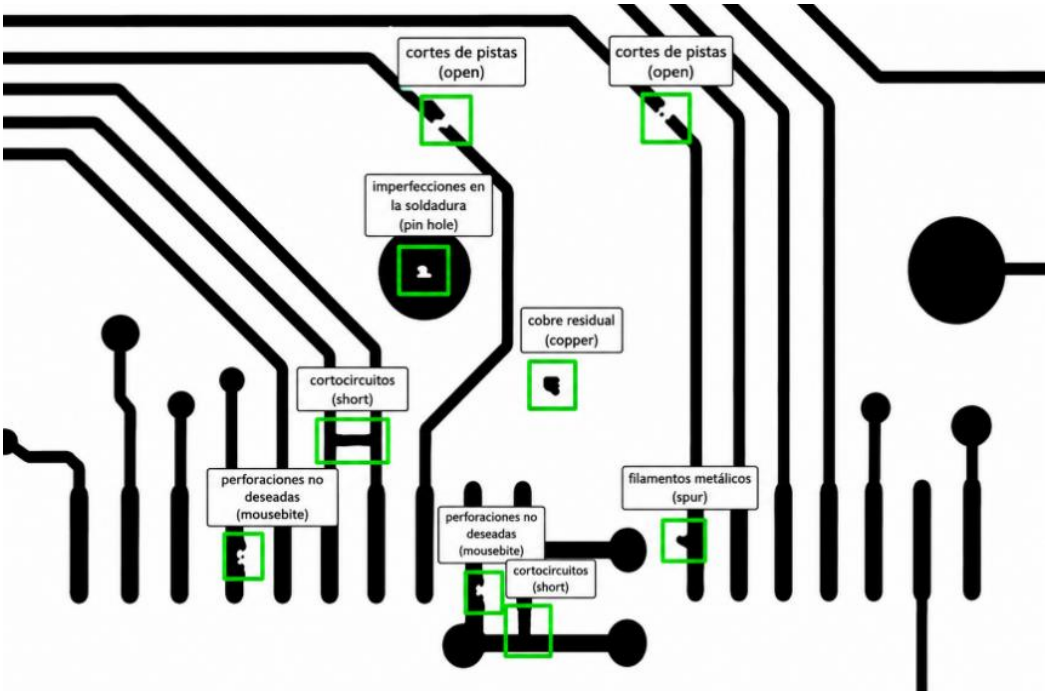

Figura 1. Placa PCB con defectos que incluyen cortes de pistas, cortocircuitos, perforaciones no deseadas, cobre residual e imperfecciones en la soldadura. Fuente: [8].

La literatura reciente evidencia un crecimiento acelerado de investigaciones orientadas a optimizar modelos YOLO para inspección de defectos en placas PCB. Por ejemplo, [17] propone utilizar YOLOv5 combinando mecanismos de atención para mejorar la extracción de características en defectos de tamaño reducido. En tanto, [19] desarrolla YOLO-CMDS, con mejoras multiescalar y refinamiento de detección para incrementar la sensibilidad del modelo frente a defectos complejos. De manera similar, [20] propone YOLO-UMS basado en fusión multiescala de características intentando detectar defectos superficiales pequeños y de baja visibilidad. Asimismo, [21] introduce YOLO-CD como una arquitectura ligera orientada a despliegues en dispositivos Edge. Por otro lado, [22] propone CSPGhost-YOLO para sistemas mixtos de detección de defectos PCB. Otras Investigaciones recientes destacan la importancia de incrementar la robustez de estos modelos en escenarios industriales complejos. En este sentido, [23] propone MDEB-YOLO que incorpora atención multiescala para micro defectos en PCB, obteniendo mejoras significativas en precisión y sensibilidad. Además, [24] desarrolla FMW-YOLO que utiliza realce frecuencial para diferenciar defectos complejos y patrones de bajo contraste.

Cabe destacar que YOLOv5 continúa siendo una de las arquitecturas más utilizadas en aplicaciones industriales debido a su estabilidad durante el entrenamiento y consistencia en métricas como precisión y sensibilidad [10], [11], [18]. No obstante, YOLOv8 incorpora modificaciones estructurales relacionadas con mecanismos de extracción multiescala, optimización de funciones de pérdida y estrategias avanzadas de entrenamiento, lo cual ha

permitido reportar mejoras en métricas como mAP@0.5 y mAP@0.5:0.95 en diversos dominios industriales. Sin embargo, en el caso de la inspección de placas de circuito impreso (PCB), la literatura y evidencia previa en el uso de YOLOv5 y YOLOv8 es más bastante limitada y dispersa. Existen trabajos que abordan la detección de defectos específicos utilizando arquitecturas personalizadas o técnicas híbridas, pero pocos realizan comparaciones directas entre las arquitecturas basadas en dichas versiones de YOLO considerando bajo condiciones de entrenamiento equivalentes. Esta brecha justificó el planteamiento metodológico del presente trabajo, centrado en evaluar el rendimiento de ambas arquitecturas. El objetivo de este estudio fue comparar el desempeño de las arquitecturas YOLOv5 y YOLOv8, en sus variantes pequeñas y medianas, para la detección automática de defectos en placas de circuito impreso, utilizando el conjunto de datos estandarizado DeepPCB, mediante la evaluación de métricas de precisión, sensibilidad, F1-score y mAP bajo distintos umbrales de IoU. La hipótesis inicial planteada en la mayoría de la literatura revisada es que las mejoras estructurales incorporadas en YOLOv8 podrían ofrecer ventajas en capacidad de detección y robustez frente a YOLOv5 para escenarios de defectos superficiales complejos sobre PCB.

2. METODOLOGÍA

El presente estudio tiene un enfoque cuantitativo y experimental, orientado a la comparación del rendimiento de arquitecturas neuronales que, como parte de la familia YOLO, se utilizan en la inspección automática de placas de circuito impreso. La revisión bibliográfica inicial permitió identificar que arquitecturas más convencionales como Faster R-CNN, ResNet, y U-Net todavía son ampliamente utilizadas en tareas de inspección visual industrial [10], [11], [13], [17]. Sin embargo, tales arquitecturas también han sido reportadas en cuanto a sus limitaciones para alcanzar tiempos de respuesta compatibles con entornos industriales. En contraste, el auge de modelos de segmentación profunda como YOLO-CMDS, YOLO-UMS, FMW-YOLO y CSPGhost-YOLO representa una tendencia actual que se orienta hacia modelos capaces de mantener altos niveles de precisión mientras logran reducir simultáneamente el costo computacional y el tiempo de inferencia, especialmente en aplicaciones PCB caracterizadas por defectos pequeños y patrones visuales complejos. Es en este ámbito donde YOLO emerge como una alternativa más eficiente para tareas de detección en tiempo real, equilibrando velocidad y precisión [25].

2.1 Procedimiento experimental

La Figura 2 presenta el flujo metodológico general implementado en la presente investigación. El procedimiento experimental fue estructurado en seis etapas principales: (i) adquisición y organización del conjunto de datos DeepPCB, (ii) preprocesamiento y adaptación de etiquetas al formato YOLO, (iii) partición estratificada de datos para entrenamiento, validación y prueba, (iv) entrenamiento de modelos YOLOv5 y YOLOv8 bajo condiciones homogéneas, (v) evaluación cuantitativa mediante métricas de detección de objetos y análisis de convergencia, y (vi) evaluación cualitativa mediante inferencia sobre imágenes no vistas.

Esta organización metodológica permite describir de forma secuencial y reproducible cada una de las fases experimentales desarrolladas. Se destacan las fases de entrenamiento, validación y prueba aplicadas sobre el conjunto de imágenes DeepPCB.

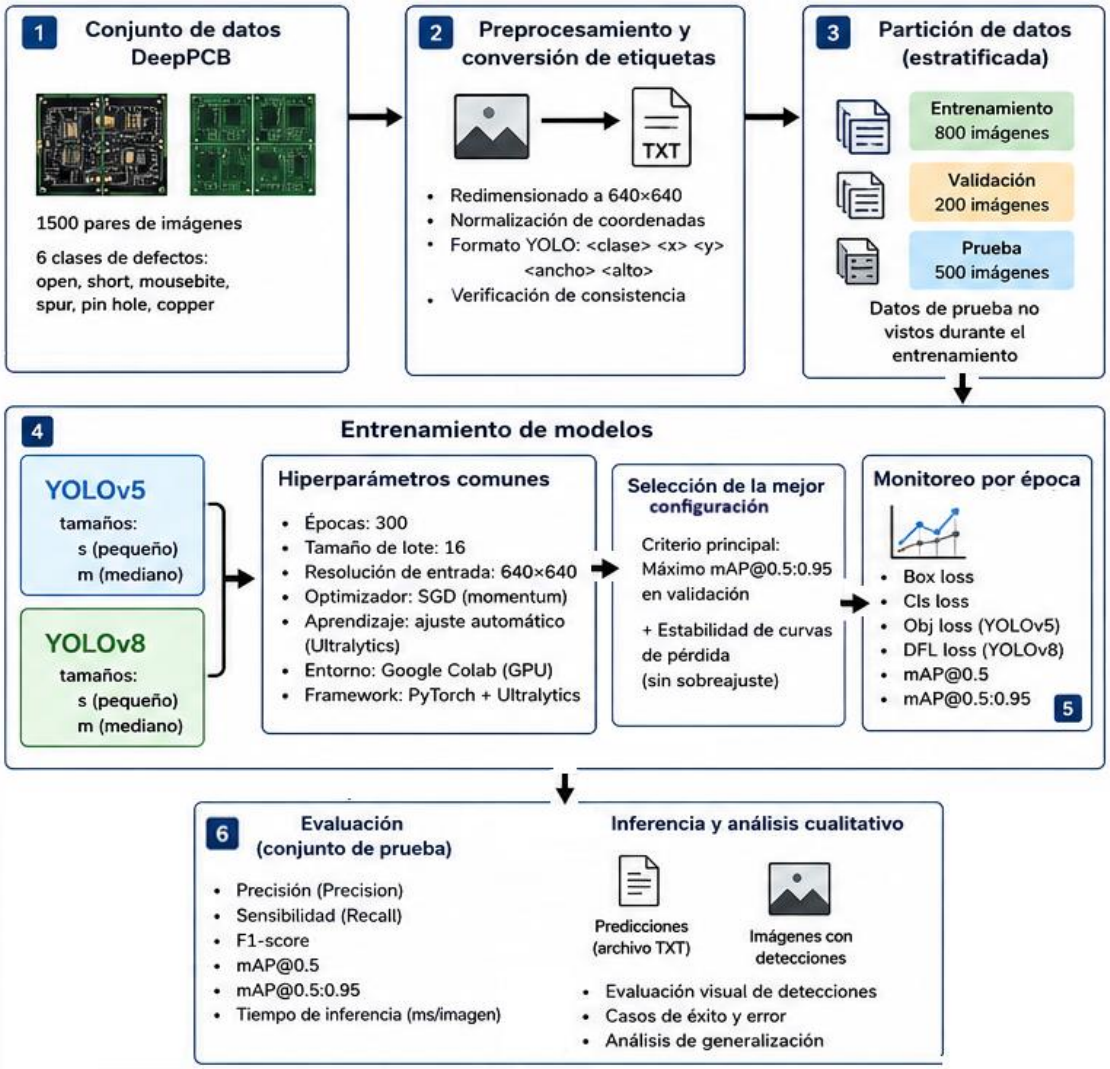

Figura 2. Esquema metodológico basado en el uso del conjunto de datos DeepPCB y modelos YOLO. Se detallan las fases de entrenamiento, validación y prueba. Fuente: elaboración propia.

2.2 Conjunto de datos deepPCB

Se trata de es un conjunto de datos público diseñado específicamente para tareas de inspección automática en placas electrónicas [8], [9]. El DeepPCB se encuentra formado por 1500 pares de imágenes obtenidas mediante un escáner lineal CCD, donde cada par está conformado por una imagen defectuosa y su correspondiente imagen de referencia sin defectos. De manera adicional, se dispone de un archivo de texto con las etiquetas de cada uno de los defectos analizados. Estas etiquetas proporcionan información sobre el defecto presente en la placa (open, short, mousebite, copper, spur y pin hole), así como su posición en la imagen capturada. Afín de garantizar condiciones experimentales reproducibles y evitar contaminación entre subconjuntos, se realizó una partición fija del conjunto de datos utilizando 800 imágenes para entrenamiento, 200 imágenes para validación y 500 imágenes para prueba. La partición se realizó de forma estratificada para preservar la distribución relativa de clases de defectos entre los subconjuntos experimentales. Asimismo, las imágenes utilizadas durante la fase de prueba permanecieron completamente aisladas del proceso de entrenamiento y ajuste de hiperparámetros, garantizando una evaluación objetiva del desempeño de generalización. El procedimiento experimental incluyó la preparación de los

subconjuntos del entrenamiento arriba mencionados, y la adaptación de los archivos que contenían las etiquetas de la imagen defectuosa en un formato compatible con la arquitectura de YOLO. Los experimentos fueron desarrollados utilizando el entorno Google Colab con aceleración GPU. El entorno computacional empleado consistió en Python 3.11, PyTorch, CUDA y las implementaciones oficiales de Ultralytics correspondientes a YOLOv5 y YOLOv8.

2.3 Consideraciones en el entrenamiento y validación de los modelos

Para garantizar comparabilidad experimental, todos los modelos fueron entrenados utilizando idénticas configuraciones de hiperparámetros: 300 épocas, tamaño de lote de 16 imágenes y resolución de entrada de 640×640 píxeles. Adicionalmente, se utilizó optimización basada en descenso de gradiente estocástico con actualización automática de tasa de aprendizaje implementada por Ultralytics. Durante el proceso de entrenamiento de cada modelo, se registró la evolución de las funciones de pérdida asociadas a la detección [26], [27]. Particularmente se consideró la pérdida de caja/recuadro (box_loss), la pérdida de clasificación (cls_loss), la pérdida de confianza en la presencia de objetos (obj_loss, en el caso de YOLOv5), y la pérdida de distribución focal (DFL, en el caso de YOLOv8). Estas funciones permiten evaluar el aprendizaje del modelo relacionados en cuanto a localización espacial de defectos, clasificación correcta, y estabilidad de convergencia durante la optimización. Con el propósito de garantizar un criterio uniforme de comparación, la selección de la mejor configuración del modelo se realizó considerando simultáneamente el valor máximo de mAP@0.5:0.95 sobre el conjunto de validación, y la estabilidad observada en las curvas de pérdida. Este criterio evita seleccionar modelos afectados por sobreajuste o inestabilidad de convergencia. Asimismo, las curvas de entrenamiento y validación fueron analizadas para identificar fenómenos asociados a sobreajuste, subajuste, y oscilaciones de optimización, proporcionando información complementaria sobre la robustez del aprendizaje alcanzado por cada arquitectura. Por ejemplo, cuando la pérdida en el conjunto de validación desciende de manera paralela a la de entrenamiento, se interpreta que el modelo está generalizando correctamente a datos no vistos. En cambio, si la pérdida de entrenamiento sigue disminuyendo mientras la de validación se estanca o incluso aumenta, esto indica que el modelo está memorizando los ejemplos en lugar de aprender patrones útiles, fenómeno que se conoce como sobreajuste. Por otro lado, la presencia de oscilaciones bruscas en las curvas, especialmente en validación, puede ser un síntoma de inestabilidad en el proceso de optimización y mayor dificultad para alcanzar un punto de operación robusto.

Para evaluar el rendimiento de los modelos de manera cualitativa, se recurrió a métricas ampliamente aceptadas en tareas de detección de objetos y visión artificial, con las que se midió tanto la detección de los defectos como la precisión en su localización. Todas estas métricas se construyen a partir de cuatro posibles resultados de las predicciones [28]: verdaderos positivos (TP), cuando el modelo detecta correctamente un defecto presente; falsos positivos (FP), cuando predice un defecto inexistente; falsos negativos (FN), cuando no detecta un defecto real; y verdaderos negativos (TN), cuando no realiza ninguna detección en una zona sin defectos. La precisión (1) mide la proporción de detecciones correctas entre todas las predicciones positivas, por lo que está directamente relacionada con la presencia de falsos positivos. Un valor alto de precisión indica que, cuando el modelo señala un defecto, normalmente acierta. La sensibilidad o recall (2), en cambio, mide la proporción de defectos detectados respecto al total de defectos presentes, y por tanto depende de los falsos negativos. En este caso, un valor alto significa que el modelo apenas deja pasar defectos sin detectar [29].

$$Precisión = \frac{TP}{TP + FP} \tag{1}$$

$$Sensibilidad = \frac{TP}{TP + FN} \tag{2}$$

El F1-score combina ambas métricas en un único valor mediante la media armónica (3). Esta métrica es útil porque permite evaluar el equilibrio entre precisión y sensibilidad, algo especialmente importante en contextos industriales. Un modelo demasiado estricto puede tener una precisión muy alta, pero fallar en cuanto a sensibilidad, dejando escapar defectos relevantes; mientras que un modelo muy permisivo puede detectar casi todo, pero generar demasiadas falsas alarmas [28]. El F1-score refleja ese balance.

$$F1\ score = \frac{2 \cdot (precisión \cdot sensibilidad)}{precisión + sensibilidad} \tag{3}$$

Finalmente, se obtuvo la precisión media promedio (mAP) (4), calculada a partir de las curvas de Precisión-Sensibilidad, lo que brinda una idea de la relación entre precisión y sensibilidad a lo largo de distintos umbrales de confianza teniendo en cuenta los valores normalizados que reflejan qué tan seguro está un modelo de la detección predicha. Se consideraron dos variantes: (i) el mAP@0.5, que evalúa la coincidencia entre la ubicación de la caja predicción y ubicación de la caja de la etiqueta real usando un umbral fijo de solapamiento (IoU ≥ 0.5), y (ii) el mAP@0.5:0.95, que promedia resultados en diez umbrales de IoU entre 0.5 y 0.95, ofreciendo una medida más estricta. Este último valor resulta especialmente relevante porque revela la capacidad del modelo no solo para detectar, sino para localizar los defectos con mayor exactitud.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{4}$$

Además de las métricas cuantitativas, se incorporó una evaluación cualitativa mediante inferencia sobre imágenes no vistas durante el entrenamiento. Para ello, se desarrolló un script de inferencia en Python capaz de cargar automáticamente los modelos entrenados y generar predicciones sobre imágenes externas al conjunto de entrenamiento. Este procedimiento permitió analizar visualmente la capacidad de generalización de cada arquitectura frente a defectos complejos y escenarios similares a condiciones industriales reales. El sistema de inferencia generó dos tipos principales de resultados. El primero corresponde a archivos estructurados en formato texto conteniendo coordenadas espaciales, clases detectadas y probabilidades asociadas a cada predicción. El segundo resultado corresponde a representaciones gráficas donde cada defecto identificado es delimitado mediante cajas de detección acompañadas de su respectiva clase y nivel de confianza. Finalmente, los resultados cuantitativos y cualitativos fueron integrados en un análisis comparativo global orientado a identificar fortalezas, limitaciones y compromisos existentes entre precisión, robustez y estabilidad de entrenamiento para cada una de las arquitecturas evaluadas.

3. RESULTADOS Y DISCUSIÓN

A fin de comparar el desempeño de las arquitecturas neuronales de cada familia YOLO, se estableció un diseño de experimentos que considera cuatro configuraciones posibles: YOLOv5s, YOLOv5m, YOLOv8s y YOLOv8m. Todas ellas fueron entrenadas bajo condiciones equivalentes y evaluadas sobre el conjunto de validación y sobre un conjunto de imágenes no vistas, lo que permitió medir su rendimiento tanto en escenarios controlados como en situaciones más próximas a la aplicación real. Este enfoque permitió garantizar comparabilidad experimental directa entre arquitecturas, evitando sesgos derivados de configuraciones de entrenamiento diferenciadas, situación frecuentemente reportada como limitación en estudios comparativos recientes sobre detección de defectos PCB.

3.1 Evolución de las funciones de pérdida

Las Figuras 3-6 muestran la evolución de las funciones de pérdida durante el entrenamiento de los modelos analizados, incluyendo las curvas de entrenamiento (fila superior) y de validación (fila inferior). Las curvas de validación son la referencia más fiable porque miden el comportamiento del modelo sobre datos no utilizados en el entrenamiento. En cambio, las de entrenamiento solo reflejan el ajuste a los ejemplos vistos y, por tanto, pueden dar una falsa sensación de mejora incluso cuando el modelo empieza a memorizar los datos (sobreajuste) o cuando no llega a aprender los patrones relevantes (subajuste). Observe que, de forma general, todos los modelos presentan una tendencia convergente durante las primeras etapas de entrenamiento, lo que indica una reducción progresiva de las funciones de pérdida asociadas a localización, clasificación y detección de objetos. Sin embargo, se perciben diferencias relevantes en la estabilidad y velocidad de convergencia entre arquitecturas YOLOv5 y YOLOv8.

En particular, las Figuras 3 y 4 muestran que YOLOv5s y YOLOv5m consiguen una disminución rápida y estable de las pérdidas de validación, que es incluso más notable en las métricas de box_loss (pérdida_recuadro) y cls_loss (pérdida_clas) que muestran oscilaciones reducidas durante gran parte del entrenamiento. Este comportamiento sugiere una adecuada capacidad de generalización y una optimización estable del proceso de aprendizaje para dichos modelos.

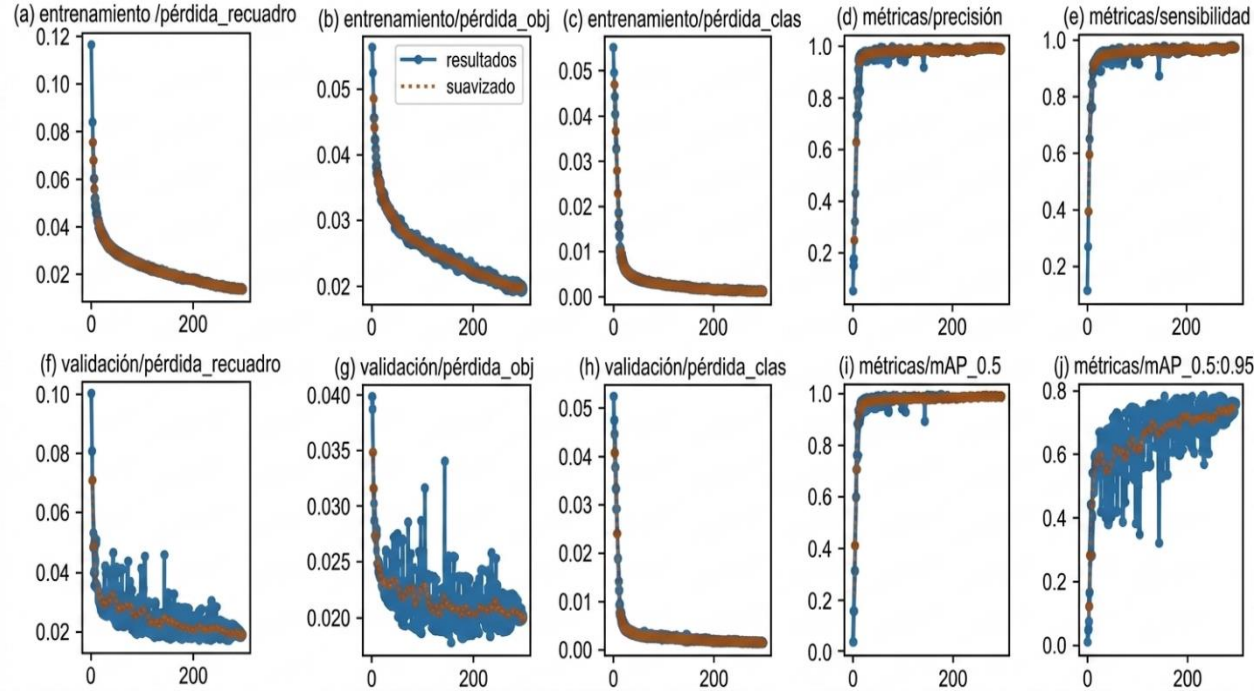

Figura 3. Evolución de las funciones de pérdida durante el entrenamiento de YOLOv5s.
Fuente: elaboración propia.

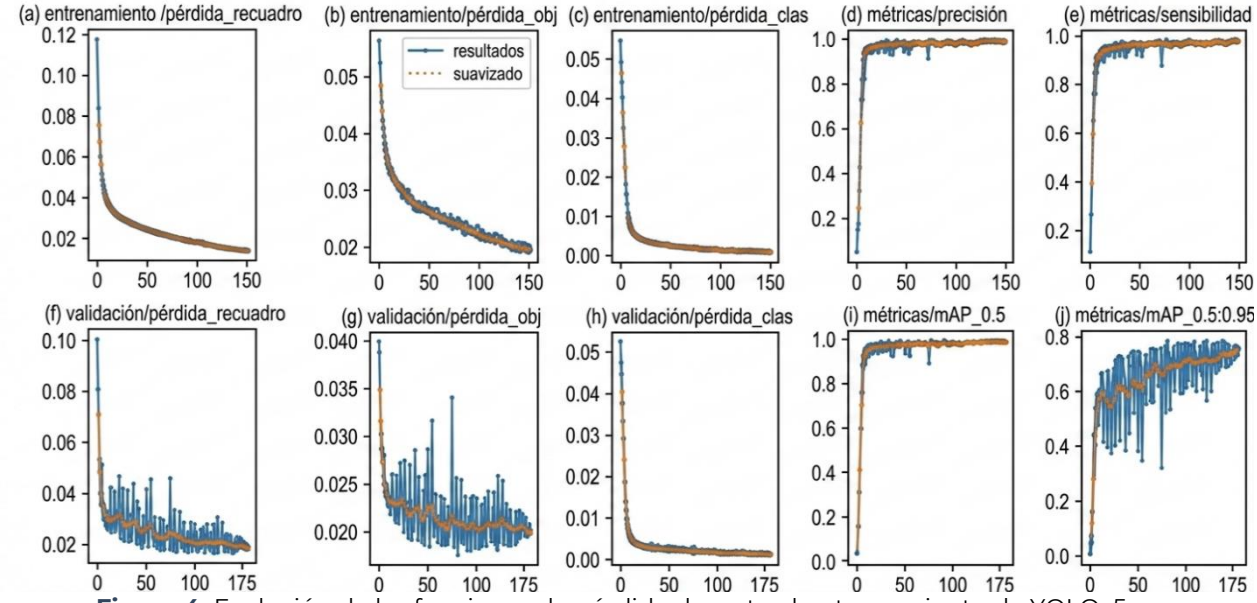

Figura 4. Evolución de las funciones de pérdida durante el entrenamiento de YOLOv5m.
Fuente: elaboración propia.

En contraste, las Figuras 5 y 6 ilustran que los modelos YOLOv8 llegan a presentar mayores fluctuaciones en ciertas etapas del entrenamiento, especialmente en la componente DFL. Si bien estas oscilaciones no implican un deterioro directo en el desempeño del modelo, sí reflejan una dinámica de optimización más compleja y sensible a la actualización de pesos. De manera similar, YOLOv8m presentó variaciones más pronunciadas en la validación durante las etapas intermedias, indicando una mayor complejidad paramétrica y sensibilidad frente a patrones visuales ambiguos presentes en defectos PCB.

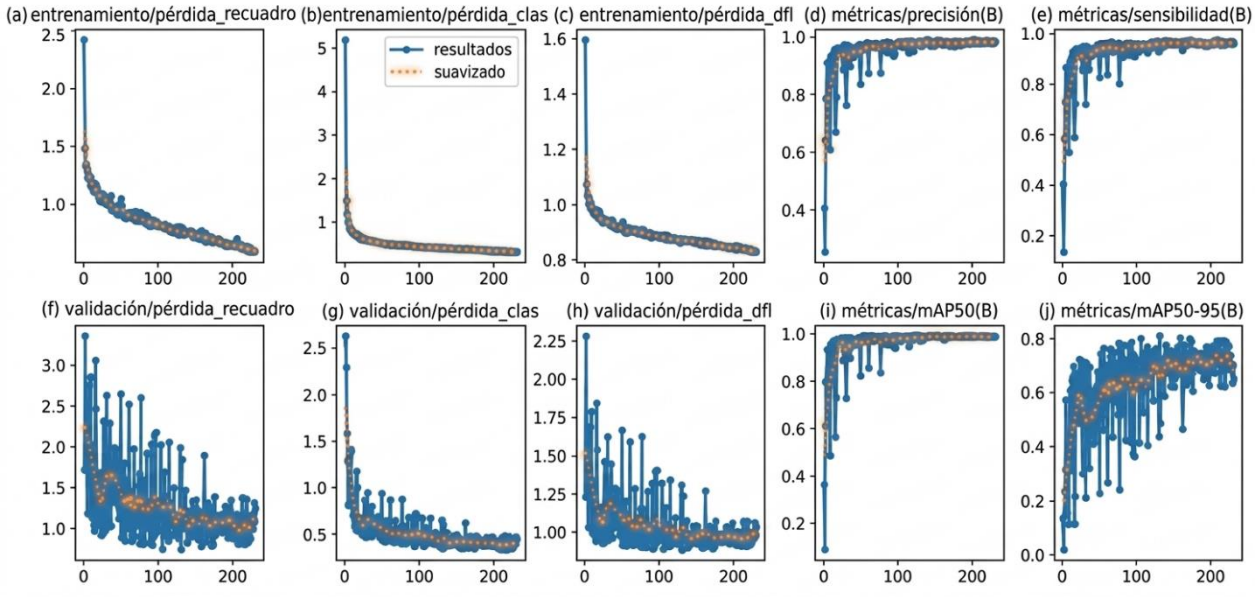

Figura 5. Evolución de las funciones de pérdida durante el entrenamiento de YOLOv8s.
Fuente: elaboración propia.

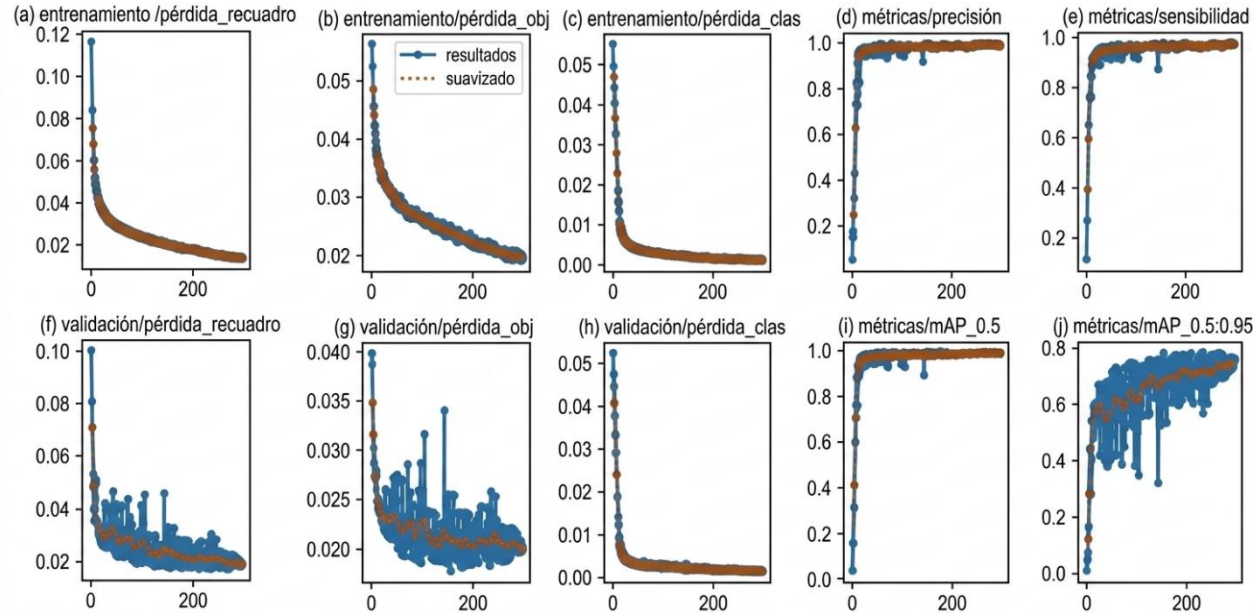

Figura 6. Evolución de las funciones de pérdida durante el entrenamiento de YOLOv8m.
Fuente: elaboración propia.

Como se ha mencionado anteriormente, la selección de la mejor época para cada modelo se realizó aplicando el criterio de fitness definido por Ultralytics, que pondera las métricas mAP@0.5 (10 %) y mAP@0.5:0.95 (90 %). Cabe destacar que este criterio prioriza la precisión espacial de localización sobre múltiples umbrales IoU, proporcionando una medida más estricta y representativa del comportamiento real de los modelos en tareas de inspección visual. Los valores de pérdida obtenidos se resumen en la Tabla 1 como punto de comparación entre configuraciones implementadas.

Tabla 1. Evolución de las funciones de pérdida durante el entrenamiento de YOLOv8m.
Fuente: elaboración propia.

	YOLOv5s	YOLOv5m	YOLOv8s	YOLOv8m
	Época 235	Época 149	Época 131	Época 69
Box loss	Muy baja (0.016966) – localiza con gran precisión	Muy baja (**0.016841**) – localización más consistente que v5s	Más alto (0.7485) – localización aceptable	Más alto (0.79148) – localización aceptable
Cls loss	Muy baja (**0.0014197**) - clasificación estable	Muy baja (0.001506) - clasificación precisa	Moderado (0.34887) - clasificación correcta en la mayoría de los casos	Moderado/alto (0.40767) - clasificación menos consistente
Dfl loss	-	-	Estable (**0.89217**) - sin sobreajuste, optimización adecuada	Oscilante (0.9211) - optimización menos estable
Obj loss	Bajo (0.018303) - detección de objetos fiable	Bajo (**0.017923**) - detección sólida y estable	-	-

* Mejores resultados resaltados en negrita

En conjunto, los resultados obtenidos señalan que YOLOv5s y YOLOv5m presentan las pérdidas de validación más bajas y estables, alcanzando valores mínimos en box_loss y cls_loss.

En ambos casos, los modelos consiguen una elevada capacidad para localizar y clasificar correctamente los defectos en las imágenes PCB. No obstante, YOLOv5m presenta una mejor convergencia respecto a su versión ligera. Por su parte, los modelos YOLOv8s y YOLOv8m, que incorporan la función de pérdida DFL en lugar de obj_loss, requieren un rango de valores distinto por diseño, por lo que no son directamente comparables con los de la serie v5. Aun así, las curvas de YOLOv8s mostraron una evolución estable y sin indicios de sobreajuste. En particular, YOLOv8m presentó mayores oscilaciones en validación, señal de una optimización menos consistente. Pese a esta inestabilidad, ambos alcanzaron un rendimiento competitivo, aunque con menor regularidad que los modelos equivalentes en la versión YOLOv5.

Los resultados obtenidos de esta comparación sugieren que, si bien YOLOv8 introduce mecanismos avanzados de localización multiescala, su comportamiento puede volverse más sensible frente a conjuntos de datos relativamente pequeños y defectos de alta similitud visual, como ocurre en DeepPCB. Mismo así, el análisis conjunto de las curvas de entrenamiento y validación permitió identificar que ninguno de los modelos presentó fenómenos severos de sobreajuste. De hecho, existe una diferencia relativamente reducida entre pérdidas de entrenamiento y validación que indica una adecuada capacidad de generalización, aspecto que es particularmente relevante en aplicaciones industriales.

3.2 Métricas de rendimiento en validación

La Tabla 2 resume la evolución de las métricas de rendimiento. Tal como se observa en las Figuras 3-7, en todos los casos se alcanzaron valores de mAP@0.5 superiores al 98 % y de mAP@0.5:0.95 en torno al 78–81 %, lo que confirma una elevada capacidad de detección y localización de defectos por parte de los modelos.

Tabla 2. Métricas de evaluación en la mejor época para cada modelo. Fuente: elaboración propia.

Modelo	Época	Precisión	Sensibilidad	mAP@0.5	mAP@0.5:0.95
YOLOv5s	235	Muy alto (0.98662) - predicciones casi siempre correctas	Muy alto (0.97215) - rara vez deja defectos sin detectar	Excelente (0.9889) - gran capacidad de detección a IoU=0.5	Bueno (0.78278) - localización robusta en varios umbrales
YOLOv5m	149	Máximo (**0.99086**) - detecciones extremadamente fiables	Muy alto (**0.97397**) - prácticamente no pierde defectos	Sobresaliente (**0.99071**) - mejor rendimiento global	Bueno (0.78325) - localización consistente y estable
YOLOv8s	131	Muy alto (0.98475) - aunque algo menor que v5m	Alto (0.96588) - detecta la mayoría de los defectos	Excelente (0.98947) - precisión muy competitiva	Muy bueno (**0.81154**) - mejor localización a diferentes IoU
YOLOv8m	69	Alto (0.96519) - fiable, pero con más falsos positivos que otros	Alto (0.95147) - pierde algunos defectos frente a v5	Muy bueno (0.9816) - buen desempeño general	Bueno (0.7821) - localización adecuada, aunque menos destacada que v8s

* Mejores resultados resaltados en negrita

No obstante, los resultados muestran que YOLOv5m fue el modelo más consistente, al combinar valores de precisión y sensibilidad muy competitivos, lo que lo convierte en la opción más equilibrada para aplicaciones industriales. En cambio, YOLOv8s destacó particularmente en la métrica mAP@0.5:0.95, la cual refleja una mejor localización en distintos umbrales de IoU, aunque con valores de precisión y sensibilidad ligeramente inferiores a los resultados

obtenidos por YOLOv5. El modelo YOLOv8m, pese a las oscilaciones observadas en el entrenamiento, logró métricas estables y cercanas a las de sus homólogos, mientras que YOLOv5s ofreció un rendimiento competitivo con menor coste computacional.

Este comportamiento podría indicar que el incremento de complejidad paramétrica presente en YOLOv8m no necesariamente se traduce en mejoras de desempeño cuando se trabaja con conjuntos de datos de tamaño moderado como DeepPCB.

3.3 Evaluación en imágenes no vistas

Para estimar el rendimiento en condiciones más próximas a un escenario real, se realizó una prueba adicional sobre 500 imágenes no vistas durante el entrenamiento, ni la validación. Cada modelo generó predicciones que fueron comparadas con las anotaciones originales, calculándose métricas estándar como precisión, sensibilidad, F1-score y mAP@0.5. Los resultados globales se presentan en la Tabla 3.

Tabla 3. Métricas de rendimiento en imágenes no vistas. Fuente: elaboración propia.

Modelo	TP	FP	FN	Precisión	Sensibilidad	F1	mAP50
YOLOv5s	3047	283	93	Alto (0.9150) - predicciones fiables, aunque algunos falsos positivos	Muy alto (0.9704) - detecta la mayoría de defectos	Excelente (0.9419) - equilibrio entre precisión y sensibilidad	Muy bueno (0.9526) - detección sólida en IoU=0.5
YOLOv5m	3055	276	85	Alto (**0.9171**) - detecciones muy fiables	Muy alto (**0.9729**) - prácticamente no pierde defectos	Excelente (**0.9442**) - mejor combinación global	Excelente (**0.9621**) - máximo desempeño en imágenes nuevas
YOLOv8s	3024	303	116	Alto (0.9089) - algo más de falsos positivos que v5	Alto (0.9631) - detecta la mayoría de defectos	Muy bueno (0.9352) - equilibrio correcto pero menor que v5	Muy bueno (0.9494) - rendimiento competitivo
YOLOv8m	2977	274	163	Alto (0.9157) - fiable pero más errores en casos concretos	Bueno (0.9481) - pierde más defectos que otros modelos	Muy bueno (0.9316) - estable, aunque menos equilibrado	Bueno (0.9349) - rendimiento correcto, pero por debajo de v5

* Mejores resultados resaltados en negrita

Los valores obtenidos confirman la coherencia con lo observado en validación: todos los modelos superan el 94 % en F1-score y alcanzan un mAP@0.5 superior al 93 %. Entre ellos, YOLOv5m volvió a destacar como la opción más equilibrada, con la mejor combinación de F1 y mAP@0.5, lo que lo posiciona como el más fiable en aplicaciones de inspección industrial. YOLOv5s obtuvo resultados muy próximos con menor coste computacional, mientras que YOLOv8s mostró un ligero descenso en precisión y sensibilidad, aunque alcanzó métricas competitivas. YOLOv8m, pese a una mayor inestabilidad en el entrenamiento, mantuvo un rendimiento estable, pero con más errores en defectos concretos. El análisis por clase presentado en la Tabla 4 refleja que las variantes de YOLOv5 ofrecen nuevamente el rendimiento más consistente para la mayoría de defectos, con valores superiores al 97 % en categorías como open, mousebite (muesca) o copper (cobre).

Tabla 4. Métrica mAP@0.5:0.95 por clase en imágenes no vistas. Fuente: elaboración propia.

Clase/modelo	YOLOv5s	YOLOv5m	YOLOv8s	YOLOv8m
open (abierto)	Muy alto (0.9662) – detección casi perfecta	Máximo (**0.9785**) – detección muy fiable	Muy alto (0.9585) – buen desempeño	Alto (0.9527) – correcto pero menor consistencia
short (corto)	Alto (**0.9111**) – buena detección, aunque con errores	Alto (0.9107) – desempeño similar a v5s	Alto (0.9087) – estable pero no sobresaliente	Moderado (0.8762) – más errores y menor fiabilidad
mousebite (muesca)	Excelente (0.9794) – detección casi perfecta	Excelente (**0.9798**) – máximo desempeño	Muy alto (0.9671) – rendimiento competitivo	Muy alto (0.9573) – aunque por debajo de v5
spur (rebaba)	Muy alto (0.9460) – detección sólida	Muy alto (**0.9493**) – desempeño consistente	Muy alto (0.9392) – competitivo pero menor	Alto (0.8995) – notable descenso de fiabilidad
copper (cobre)	Excelente (**0.9786**) – detección muy precisa	Excelente (0.9784) – máximo desempeño	Excelente (0.9775) – prácticamente igual a v5	Muy alto (0.9694) – sólido, aunque menor
Pinole (orificio)	Muy alto (0.9346) – detección buena y estable	Excelente (**0.9758**) – máximo en esta clase	Muy alto (0.9451) – correcto, aunque menor	Muy alto (0.9545) – desempeño competitivo

* Mejores resultados resaltados en negrita

En contraste, YOLOv8m muestra un descenso más notable en clases como spur (rebaba) y short (corto), lo que indica una mayor sensibilidad a falsos positivos y dificultades para distinguir defectos similares. Este comportamiento podría estar relacionado con la sensibilidad de las arquitecturas más profundas frente a pequeñas variaciones geométricas presentes en defectos PCB, especialmente cuando existen regiones visuales altamente repetitivas o bajo contraste entre pistas conductoras. Siendo así, la combinación de precisión, sensibilidad y estabilidad observada en YOLOv5m sugiere que esta arquitectura puede conseguir un balance más adecuado en cuanto a representatividad y complejidad computacional, lo que le permite generalizar correctamente sobre patrones visuales no observados previamente. En cuanto a comparación interna, observe que YOLOv5s obtiene resultados muy próximos a los alcanzados por YOLOv5m, aunque con una ligera disminución en precisión y mAP@0.5. Sin embargo, considerando su menor complejidad computacional, dicho modelo continúa representando una alternativa particularmente atractiva para aplicaciones donde el tiempo de inferencia y el consumo de recursos constituyen restricciones importantes.

Finalmente, en las Figuras 7-9 se brindan ejemplos representativos de las inferencias que tienen lugar sobre imágenes no vistas, y que permiten contrastar los resultados numéricos con observaciones cualitativas. Estas visualizaciones ponen de manifiesto que los modelos difieren en su capacidad para detectar defectos pequeños o complejos y que, a su vez, los errores más frecuentes están relacionados con omisiones y confusión entre clases. Tales errores fueron más frecuentes en las variantes YOLOv8, particularmente en YOLOv8m, mientras que YOLOv5m mantuvo un comportamiento más estable y consistente.

Con base en estos resultados, es posible concluir que todas las arquitecturas evaluadas presentan un elevado potencial para aplicaciones de inspección visual automatizada en PCB. Sin embargo, considerando conjuntamente estabilidad de entrenamiento, métricas cuantitativas, capacidad de generalización y comportamiento cualitativo, YOLOv5m emerge como la configuración más equilibrada para escenarios de inspección industrial de placas PCB.

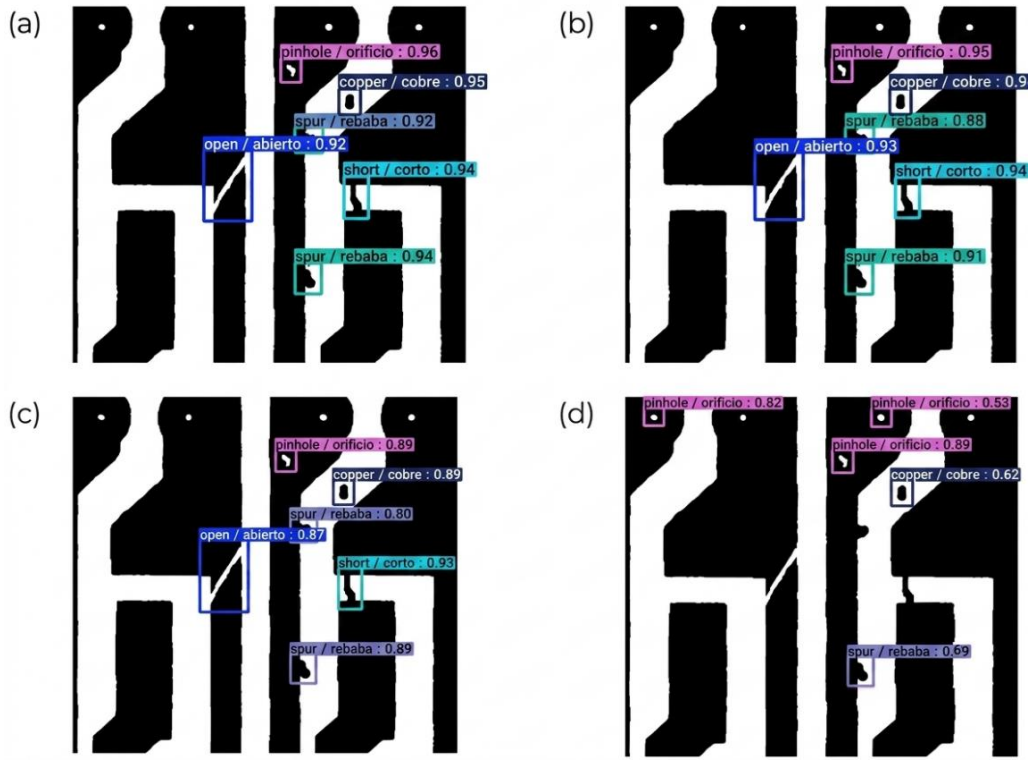

Figura 7. Resultados de inferencia en la imagen 92000116 con los cuatro modelos: (a) YOLOv5s, (b) YOLOv5m, (c) YOLOv8s, (d) YOLOv8m. Fuente: elaboración propia.

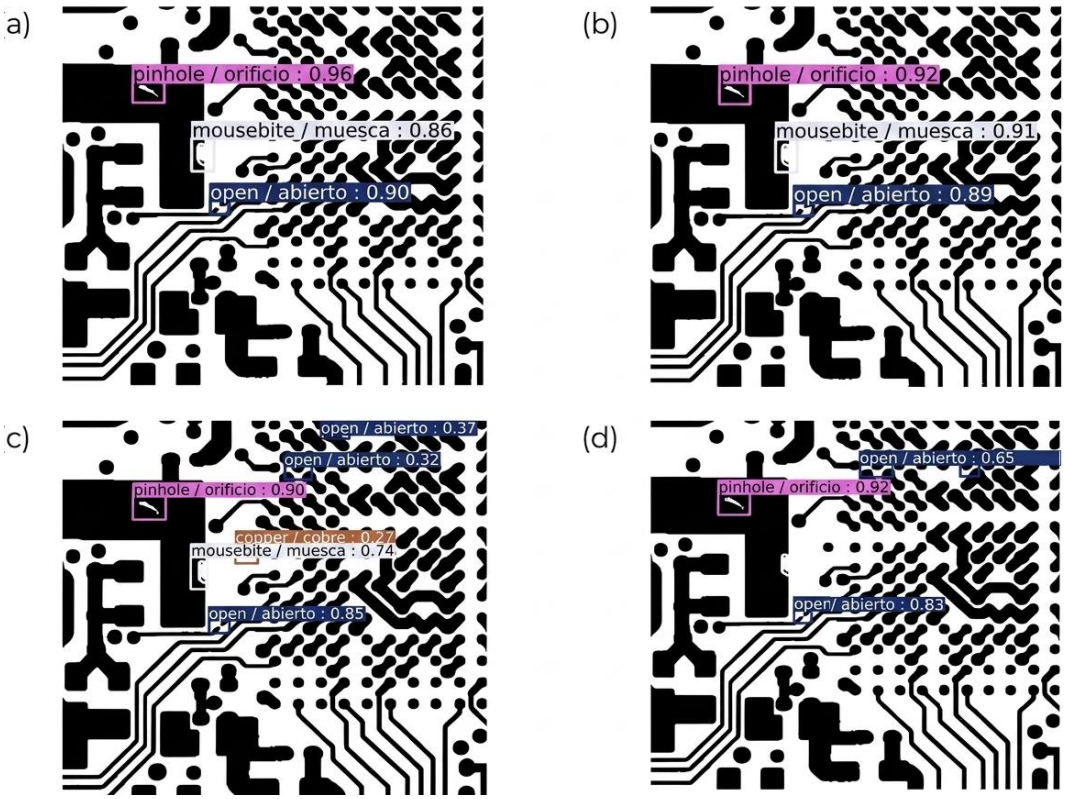

Figura 8. Resultados de inferencia en la imagen 90100028 con los cuatro modelos: (a) YOLOv5s, (b) YOLOv5m, (c) YOLOv8s, (d) YOLOv8m. Fuente: elaboración propia.

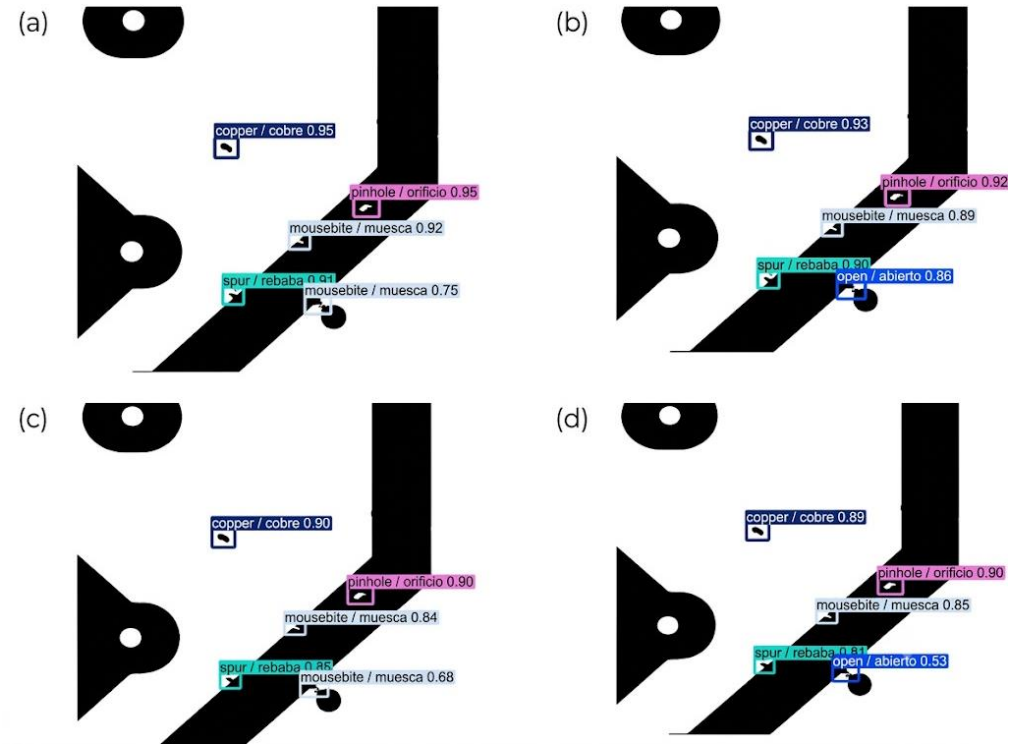

Figura 9. Resultados de inferencia considerando los cuatro modelos comparados: (a) YOLOv5s, b) YOLOv5m, (c) YOLOv8s, (d) YOLOv8m. Fuente: elaboración propia.

3.4 Discusión

Los resultados experimentales obtenidos son consistentes con la bibliografía más reciente, que señala a las arquitecturas de la familia YOLO como una opción eficiente para el reconocimiento de defectos en placas PCB [10], [11], [17], [18]. En comparación con variantes modificadas mediante mecanismos de atención y fusión multiescala (YOLO-CMDS, YOLO-UMS, YOLO-CD, CSPGhost-YOLO, MDEB-YOLO y FMW-YOLO) [19]–[24], el presente estudio se centra exclusivamente en las arquitecturas oficiales YOLOv5 y YOLOv8 bajo condiciones experimentales homogéneas. excluyendo mejoras por cambios en la arquitectura o tácticas específicas de entrenamiento. En cuanto a resultados, YOLOv8s alcanzó el mejor desempeño en mAP@0.5:0.95, con mayor acierto al ubicar espacialmente los defectos en las placas PCB. No obstante, YOLOv5m tuvo un desempeño global superior en imágenes no vistas. Los resultados obtenidos sugieren que el desempeño global del modelo no se ve afectado durante la detección de defectos en placas PCB, a pesar de las mejoras estructurales implementadas en YOLOv8. Esta observación coincide con investigaciones recientes [1]–[3], [10], las cuales resaltan que arquitecturas con mayor complejidad no siempre ofrecen ventajas ante defectos de alta similitud visual. Este estudio, en comparación con otras investigaciones, evidencia que las implementaciones oficiales de YOLO siguen brindando un rendimiento competitivo cuando son entrenadas en condiciones experimentales controladas. Desde una perspectiva industrial, estos resultados indican que es posible reducir la complejidad computacional sin comprometer la precisión en la detección de defectos en placas PCB.

4. CONCLUSIONES

Este estudio comparó cuatro configuraciones de la familia YOLO en la inspección automática de defectos en placas de circuito impreso, utilizando el mismo protocolo de

entrenamiento y evaluación sobre el conjunto de datos DeepPCB, junto con pruebas adicionales con imágenes no vistas. Los resultados muestran que YOLOv5m ofrece el mejor rendimiento global al combinar alta precisión, sensibilidad, y estabilidad en la validación y menor incidencia de falsos positivos, lo que lo convierte en la opción más adecuada para entornos industriales donde la fiabilidad resulta prioritaria. Muy cerca se sitúa YOLOv5s, que, con un menor coste computacional, representa una alternativa práctica cuando existen restricciones de hardware. En el caso de YOLOv8, aunque demostró buen desempeño en métricas exigentes de localización, no logró superar a los modelos de la serie YOLOv5 en pruebas de generalización. Si bien YOLOv8m logró resultados competitivos, también es cierto que mostró un comportamiento más irregular respecto al conjunto evaluado. Los resultados alcanzados no confirman la hipótesis inicial respecto a la superioridad de YOLOv8. Sin embargo, teniendo en cuenta el contexto específico de inspección de PCB con DeepPCB y el régimen de entrenamiento aplicado, la arquitectura YOLOv5m si resulta ser la opción más robusta al equilibrar detección, localización y estabilidad. En este sentido, YOLOv5s se perfila como una alternativa eficiente cuando se prioriza latencia o coste computacional. En cuanto a trabajos futuros podría ser conveniente ampliar la base de datos con imágenes procedentes de entornos industriales, y validar los modelos directamente en procesos de fabricación considerando también para cada modelo la estabilidad en operación continua, los tiempos de inferencia y su impacto en la reducción de defectos no detectados.

5. AGRADECIMIENTO Y FINANCIACIÓN

Los autores agradecen el apoyo financiero de FAPERJ, Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro (Brasil), Pós-Doutorado Sênior (PDS), Edital-FAPERJ N° 18/2024 (Processo E-26/200.560/2025).

6. REFERENCIAS

[1] Q. Ling, and N. A. M. Isa, “Printed Circuit Board Defect Detection Methods Based on Image Processing, Machine Learning and Deep Learning: A Survey,” *IEEE Access*, vol. 11, pp. 15921-15944, Feb. 2023. http://doi.org/10.1109/ACCESS.2023.3245093

[2] Y. Zhou, M. Yuan, J. Zhang, G. Ding, and S. Qin, “Review of vision-based defect detection research and its perspectives for printed circuit board,” *J. Manuf. Syst.*, vol. 70, pp. 557–578, Oct. 2023. https://doi.org/10.1016/j.jmsy.2023.08.019

[3] S. Alawandi, K. Mallibhat, U. Kudachi, and A. Beedanal, “PCB defects: A unified survey of trends, detection techniques, and limitations through systematic literature review,” *J. Electron. Test.*, vol. 41, pp. 709-787, Dec. 2025. https://doi.org/10.1007/s10836-025-06198-y

[4] X. Yu, H. Sun, Y. He, J. Peng, X. Zeng, and L. Chen, “Robust classification method for printed circuit board defects based on virtual and real space cooperative diffusion model,” *Eng. Appl. Artif. Intell.*, vol. 169, p. 114126, Apr. 2026. https://doi.org/10.1016/j.engappai.2026.114126

[5] C. Zhang, G. Shi, H. Li, M. Yang, Y. Li, and Z. Bing, “Circuit board welding defect detection based on industrial IoVT,” *IEEE Internet Things J.*, vol. 13, no. 7, pp. 14003-14018, Apr. 2026. https://doi.org/10.1109/JIOT.2026.3651257

[6] Fineline, “Proceso de fabricación de PCB – Fineline Global,” fineline-global.com. Accessed: Mar. 21, 2026. [Online]. Available: https://www.fineline-global.com/es/knowledge-centre/pcb-manufacturing-process/

[7] PCBWay Assitant 03, “Proceso de producción y nuestros equipos,” pcbway.es. Accessed: Mar. 21, 2026. [Online]. Available: https://www.pcbway.es/pcb-service.html?step=10

[8] S. Tang, “tangsanli5201/DeepPCB,” GitHub.com. Accessed: Feb. 21, 2026. [Online]. Available: https://github.com/tangsanli5201/DeepPCB

[9] S. Lv *et al*., “A dataset for deep learning based detection of printed circuit board surface defect,” *Sci. Data*, vol. 11, p. 811, Jul. 2024. https://doi.org/10.1038/s41597-024-03656-8

[10] X. Chen, Y. Wu, X. He, and W. Ming, "A comprehensive review of deep learning-based PCB defect detection," *IEEE Access*, vol. 11, pp. 139017-139038, Dec. 2023. https://doi.org/10.1109/ACCESS.2023.3339561

[11] X. Wu, Y. Ge, Q. Zhang, and D. Zhang, "PCB defect detection using deep learning methods," in *IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD),* Dalian, China, 2021, pp. 873-876. http://doi.org/10.1109/CSCWD49262.2021.9437846

[12] Z. He, Y. Wu, Y. Lv, and Y. He, "PCB-Faster-RCNN: An Improved Object Detection Algorithm for PCB Surface Defects," *Appl. Sci.*, vol. 15, no. 24, p. 12881, Dec. 2025. https://doi.org/10.3390/app152412881

[13] Z. Li, and X. Liu, "Soldering Defect Segmentation Method for PCB on Improved UNet," *Appl. Sci.*, vol. 14, no. 16, p. 7370, Aug. 2024. https://doi.org/10.3390/app14167370

[14] Z. Gao, Y. Li, and S. Yuan, "DECNet: An efficient improved model for printed circuit board surface defect detection," *Signal, Image Video Process.*, vol. 20, no. 4, p. 188, Mar. 2026. https://doi.org/10.1007/s11760-026-05175-y

[15] F. Jiang, H. Chen, S. Wei, and C. Chen, "Performance Analysis of Printed Circuit Board Defect Detection with Hybrid CNN Feature Extraction and Clustering," *Eng*, vol. 7, no. 1, p. 41, Jan. 2026. https://doi.org/10.3390/eng7010041

[16] J. Shi, J. Yang, and Y. Zhang, "Research on Steel Surface Defect Detection Based on YOLOv5 with Attention Mechanism," *Electronics*, vol. 11, no. 22, p. 3735, Nov. 2022. https://doi.org/10.3390/electronics11223735

[17] Z. Yang, D. Li, L. Hou, and W. Nai, "A Comprehensive Performance Evaluation of YOLO Series Algorithms in Automatic Inspection of Printed Circuit Boards," *Machines*, vol. 14, no. 1, p. 94, Jan. 2026. https://doi.org/10.3390/machines14010094

[18] V. Adibhatla, H.-C. Chih, C.-C. Hsu, J. Cheng, M. F. Abbod, and J.-S. Shieh, "Defect detection in printed circuit boards using You-Only-Look-Once convolutional neural networks," *Electronics*, vol. 9, no. 9, p. 1547, Sep. 2020. https://doi.org/10.3390/electronics9091547

[19] J. Xu, L. Shi, W. Yang, X. Zhu, J. Li, and Y. Wang, "YOLO-CMDS: An enhanced method for PCB defect detection," *Integration*, vol. 109, p. 102732, Jul. 2026. https://doi.org/10.1016/j.vlsi.2026.102732

[20] H. Peng, W. Yang, and B. Yu, "YOLO-UMS: Multi-Scale Feature Fusion Based on YOLO Detector for PCB Surface Defect Detection," *Sensors*, vol. 26, no. 2, p. 689, Jan. 2026. https://doi.org/10.3390/s26020689

[21] Z. Wang, P. Yuan, and X. Zhang, "YOLO-CD: A lightweight real-time PCB defect detection model for edge deployments," *J. Real-Time Image Proc.*, vol. 23, p. 46, Jan. 2026. https://doi.org/10.1007/s11554-025-01847-z

[22] V. Kumar Ancha, V. Gonuguntla, and R. Vaddi "CSPGhost-YOLO: A lightweight and robust model for real-time mixed PCB defect detection system," *Measurement*, vol. 266, p. 120521, Mar. 2026. https://doi.org/10.1016/j.measurement.2026.120521

[23] X. Zuo, N. Zhao, K. Wang, and J. Hu, "MDEB-YOLO: A Lightweight Multi-Scale Attention Network for Micro-Defect Detection on Printed Circuit Boards," *Micromachines*, vol. 17, no. 2, p. 192, Jan. 2026. https://doi.org/10.3390/mi17020192

[24] Y. Li *et al.*, "FMW-YOLO: A Frequency-Enhanced and Multi-Scale Context-Aware Framework for PCB Defect Detection," *Micromachines*, vol. 17, no. 5, p. 531, Apr. 2026. https://doi.org/10.3390/mi17050531

[25] S. Singh, "Leveraging YOLO object detection for accurate and efficient visual recognition," labellerr.com. Accessed: Jun. 21, 2026. [Online]. Available: https://www.labellerr.com/blog/why-is-the-yolo-algorithm-important/

[26] G. Jocher, "Ultralytics. Box, cls, and dfl loss gain #10375," github.com. Accessed: Jun. 21, 2026. [Online]. Available: https://github.com/ultralytics/ultralytics/issues/10375

[27] P. García Mesa, "Understanding YOLOv5 Loss: A Comprehensive Analysis," medium.com. Accessed: Jun. 21, 2026. [Online]. Available: https://pgmesa.medium.com/understanding-yolov5-loss-a-comprehensive-analysis-ffe35e6203e0

[28] G. Naidu, T. Zuva, and E. M. Sibanda, "A review of evaluation metrics in machine learning algorithms," *Artificial Intelligence Application in Networks and Systems*, R. Silhavy and P. Silhavy, Eds., Cham, Switzerland: Springer, 2023, pp. 15-25. https://doi.org/10.1007/978-3-031-35314-7_2

[29] Ultralytics, "Performance Metrics Deep Dive," ultralytics.com/. Accessed: Jun. 21, 2026. [Online]. Available: https://docs.ultralytics.com/guides/yolo-performance-metrics/

CONFLICTO DE INTERÉS

Ninguno de los autores manifestó la existencia de posibles conflictos de intereses que debieran ser declarados en relación con este artículo.

CONTRIBUCIÓN DE AUTORÍA

Alexia Blanco Pila: Investigación, software, redacción, borrador Original.

Ivón O. Benítez-González: Conceptualización, curación de datos, análisis formal, metodología, verificación, visualización, redacción, revisión, y edición.

José M. Bernal-de Lázaro: Conceptualización, curación de datos, análisis formal, metodología, supervisión, verificación, visualización, redacción, revisión, y edición.