Universidad CENFOTEC

Maestría en Ingeniería de Software con énfasis en Inteligencia Artificial

Documento final de Proyecto de Investigación Aplicada 1

Tema:

Optimización de Modelos de IA para el Reconocimiento de Patrones en Tiempo Real en Sistemas Embebidos de Bajo Costo en Edge Computing

Elaborado por:

Sergio Oviedo Seas

Profesor:

Miguel Pérez Montero

Fecha: Abril, 2025

Declaratoria de Derechos de Autor

Yo, **Sergio Oviedo Seas**, autor de la tesis titulada *Optimización de Modelos de IA para el Reconocimiento de Patrones en Tiempo Real en Sistemas Embebidos de Bajo Costo en Edge Computing*, declaro que este trabajo es original y de mi autoría. Todas las fuentes utilizadas han sido debidamente citadas conforme a los lineamientos establecidos en la séptima edición del Manual de Publicación de la American Psychological Association (APA, 2020) y las normativas institucionales vigentes.

Autorizo a la **Universidad CENFOTEC** a permitir la consulta y uso de esta tesis exclusivamente con fines académicos. Queda estrictamente prohibida su reproducción total o parcial con fines comerciales o cualquier otro propósito ajeno a la actividad académica sin mi autorización previa y por escrito.

Referencia a la Declaratoria:

Oviedo Seas, S. (2025). *Optimización de Modelos de IA para el Reconocimiento de Patrones en Tiempo Real en Sistemas Embebidos de Bajo Costo en Edge Computing* [Tesis de maestría, Universidad CENFOTEC].

Dado en **Heredia**, a los **18 días del mes de abril de 2025**.

Firma:
Sergio Oviedo Seas

Tabla de Contenido

Tabla de Figuras

Índice de Tablas

Resumen Ejecutivo

En los últimos años, el desarrollo de inteligencia artificial embebida ha experimentado un notable avance, particularmente en el uso de modelos optimizados para ejecutarse en el borde (edge computing). Sin embargo, a pesar de los progresos tecnológicos, no existe un marco metodológico accesible, estructurado y validado que oriente la implementación completa de soluciones de visión por computadora en dispositivos de bajo costo. Esta falta de guía integral dificulta la adopción práctica de estas tecnologías en contextos donde los recursos computacionales y económicos son limitados, como la educación técnica, el prototipado rápido o proyectos de innovación aplicada.

La presente investigación busca responder a esa necesidad mediante el diseño de un marco metodológico modular y replicable, que permita integrar modelos de aprendizaje automático en microcontroladores como el Nicla Vision y el Portenta H7. Este marco aborda todo el proceso: desde la selección del hardware y la optimización del modelo, hasta su despliegue y validación funcional en el borde, sin depender de procesamiento en la nube. La propuesta se alinea con las tendencias globales en edge AI y TinyML, y tiene el potencial de fortalecer la formación práctica en inteligencia artificial embebida, así como fomentar la autonomía y eficiencia energética en el desarrollo de soluciones inteligentes.

Esta investigación se desarrolla en el contexto académico, bajo un enfoque técnico-aplicado, y busca generar un aporte metodológico sustentado en revisión científica, observación técnica y fundamentación tecnológica. Su finalidad no es únicamente resolver una problemática puntual, sino también ofrecer una herramienta de referencia para estudiantes, docentes y desarrolladores interesados en la aplicación real de inteligencia artificial en sistemas embebidos de bajo consumo.

Palabras Clave: optimización de algoritmos, reconocimiento de patrones, procesamiento descentralizado, computación en el borde, sistemas embebidos, IA en tiempo real, aprendizaje profundo, hardware de bajo costo.

Capítulo 1. Introducción

1.1 Generalidades

El reconocimiento de patrones es un área de la inteligencia artificial y el aprendizaje automático que se enfoca en la identificación de regularidades y estructuras en conjuntos de datos. Se utiliza en diversas aplicaciones, como visión por computadora, procesamiento de señales y sistemas de toma de decisiones.

El presente proyecto de investigación, aborda el desarrollo de un marco metodológico para la identificación de patrones en sistemas embebidos de *edge computing*, con un enfoque en *deep learning* y *machine learning* aplicados a hardware de bajo costo. La validación experimental se centra en el caso de uso de "sumo bot"[1], utilizando dispositivos como Nicla Vision y Portenta H7 Vision Bundle para el procesamiento en el borde. Los datos utilizados en esta investigación fueron obtenidos por el autor del trabajo de graduación, quien desarrolló un *dataset* específico para el caso de uso, empleando un dojo[2] de 80 cm de diámetro y robots fabricados en Costa Rica con dimensiones de 10x10 cm.

El *dataset* generado, se encuentra compuesto por 1,107 imágenes capturadas en diferentes condiciones, con el objetivo de mejorar la capacidad del sistema para generalizar en diversos escenarios. Para la captura de imágenes, se utilizó una webcam Streamplify FHD 60FPS, posicionada en un ángulo cenital fijo sobre el área de competencia. Sin embargo, para introducir variabilidad en los datos sin alterar la perspectiva, la altura de la cámara fue modificada en distintas sesiones de captura, permitiendo simular diferentes condiciones de uso.

Las imágenes se capturaron en formato RGB, con una profundidad de color de 8 bits y una resolución de 640 × 480 píxeles. Además, cuentan con una densidad de 96 DPI tanto en altura como en anchura y utilizan el perfil de color sRGB IEC61966-2.1.

El *dataset* fue diseñado para representar fielmente los escenarios que se pueden presentar en una competencia de sumo bot. Para ello, se aseguraron condiciones diversas en la iluminación y se trabajó con reflejos variados, de manera que el modelo pudiera adaptarse a entornos reales con diferentes fuentes de luz. En

[1] Sumo bot es una competencia que se lleva a cabo anualmente en la Universidad CENFOTEC y que tiene como finalidad encuentros de robots que "pelean" en una superficie para ver quién sale primero del tablero de juego.
[2] Un dojo es el tablero con un círculo pintado en blanco que delimita el área de juego del enfrentamiento de robots.

cuanto a la composición de las imágenes, el 19% corresponde exclusivamente al dojo, sin la presencia de robots, mientras que el 81% restante contiene tanto el dojo como los robots, permitiendo que el modelo aprenda a reconocerlos en distintas posiciones y situaciones.

De las 1,107 imágenes capturadas, se seleccionaron 207 para el proceso de etiquetado, el cual se llevó a cabo utilizando la herramienta Roboflow. En esta fase, se asignaron etiquetas a los elementos clave de cada imagen, asegurando que el modelo pueda identificar correctamente tanto el dojo como los robots en el entorno de competencia.

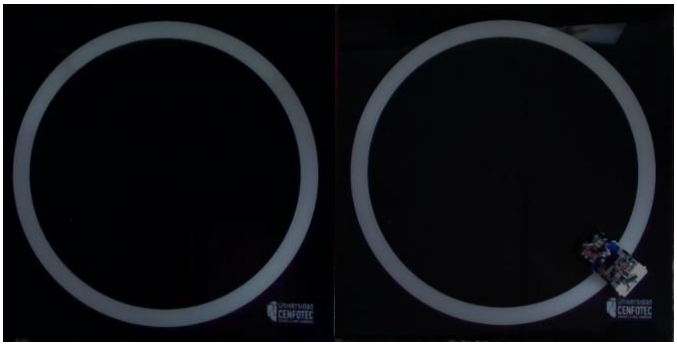

Figura 1. Imágenes del dojo y los robots sin etiquetar. Fuente: elaboración propia

Figura 2. Imágenes de los robots etiquetados. Fuente elaboración propia. Elaborado usando la aplicación Roboflow

Antes de ser utilizadas en el entrenamiento del modelo, las imágenes fueron revisadas cuidadosamente para garantizar su calidad y relevancia. Se decide conservar los valores originales de captura en cuanto a resolución y perfil de color, sin aplicar modificaciones o transformaciones adicionales, con el fin de mantener la fidelidad de los datos en un entorno de procesamiento optimizado para hardware embebido de bajo costo

Este concepto de bajo costo, en el contexto de esta investigación, no se refiere únicamente al precio absoluto del hardware, sino a una combinación de factores que optimizan la implementación de inteligencia artificial en sistemas embebidos con *edge computing*, considerando:

a) Eficiencia computacional: Dispositivos con procesadores optimizados para ejecutar modelos de IA con consumo mínimo de recursos.
b) Accesibilidad: Hardware embebido con capacidades de IA que sea viable en términos de costo-beneficio dentro del ecosistema de Edge AI.
c) Consumo energético reducido: Dispositivos que operan con baja demanda de energía para optimizar aplicaciones en tiempo real.
d) Capacidad de procesamiento en el borde: Hardware que permite inferencia sin depender de servidores externos, priorizando eficiencia en el borde.

A continuación, un comparativo de precio y características, utilizado para la toma de decisiones del hardware a utilizar:

Dispositivo	Precio Aproximado (USD)	Procesador	Cámara	Capacidad de IA	Soporte para Edge Computing	Consumo Energético
Nicla Vision	$100 - $130	Cortex-M7 (dual-core)	✅ Integrada (2MP)	✅ TensorFlow Lite, OpenMV	⚠ Limitado para modelos grandes	✅ Muy bajo
Portenta H7 Vision Bundle	$130 - $180	Cortex-M7 + Cortex-M4 (dual-core)	✅ Integrada (5MP)	✅ TensorFlow Lite, OpenMV	✅ Puede ejecutar modelos optimizados	✅ Bajo
ESP32-CAM	$10 - $30	Tensilica LX6 (dual-core)	✅ Integrada (OV2640)	❌ No apto para IA avanzada	❌ No recomendado para Edge AI	✅ Muy bajo
Jetson Nano + Módulo de Cámara	$160 - $260	Quad-core Cortex-A57 + GPU Maxwell	❌ No, requiere módulo	✅ CUDA, TensorRT, OpenCV, YOLO	✅ Excelente para Edge AI	⚠ Medio
Raspberry Pi 4 + Pi Camera	$60 - $120	Quad-core Cortex-A72	❌ No, requiere módulo	✅ TensorFlow Lite, OpenCV	✅ Moderado para Edge AI	⚠ Medio

Tabla 1. Comparativa de precios de hardware embebido. Fuente: Elaboración propia. Elaborado usando la aplicación: chatgpt

Dispositivo	Procesador	Cámara Integrada	Capacidad de IA	Soporte de OpenCV/YOLO	Notas
Nicla Vision	Cortex-M7 (dual-core)	✅ Sí, integrada (2MP)	✅ TensorFlow Lite, OpenMV	⚠ Limitado, no ideal para YOLO	Compacto, bajo consumo, especializado en TinyML.
Portenta H7 Vision Bundle	Cortex-M7 + Cortex-M4 (dual-core)	✅ Sí, integrada (5MP)	✅ TensorFlow	✅ Soporte para OpenCV y	Más potente que Nicla Vision, con

Dispositivo	Procesador	Cámara Integrada	Capacidad de IA	Soporte de OpenCV/YOLO	Notas
			Lite, OpenMV	modelos optimizados	mejor cámara integrada.
ESP32-CAM	Tensilica LX6 (dual-core)	✅ Sí, integrada (OV2640)	⚠️ Muy limitado para IA	❌ No, pero soporta procesamiento básico de imágenes	Muy barato, pero sin capacidad real de IA.
Jetson Nano	Quad-core Cortex-A57 + GPU Maxwell	❌ No, requiere módulo externo	✅ CUDA, TensorRT, TensorFlow Lite	✅ Excelente para OpenCV y YOLO	Más potente, pero requiere más energía y accesorios.
Raspberry Pi 4 + Pi Camera	Quad-core Cortex-A72	❌ No, requiere módulo externo	✅ TensorFlow Lite, OpenCV	✅ Sí, pero menos optimizado que Jetson	Solución flexible, pero requiere configuración.

Tabla 2. Comparativa de características de hardware embebido. Fuente: Elaboración propia. Elaborado usando la aplicación: chatgpt

Con esta definición, Nicla Vision y Portenta H7 Vision Bundle califican como hardware accesible para IA en *edge computing*, en contraste con soluciones más costosas y de mayor consumo como Jetson Nano o Raspberry Pi.

Este proyecto de investigación cuenta con datos de dominio público y no está sujeto a ninguna cláusula de confidencialidad. Se utiliza información técnica obtenida de fuentes abiertas y estudios previos.

1.2 Antecedentes del Problema

Durante la década de 1960, los estudios pioneros en el campo del reconocimiento de patrones establecieron los fundamentos teóricos de esta disciplina. Un aporte significativo fue el trabajo de Widrow (1964), quien propuso el uso de métodos de aprendizaje adaptativo, como el modelo ADALINE, para la identificación de patrones en datos numéricos y señales, abriendo paso a futuras aplicaciones en clasificación y control automático.

Inicialmente, el reconocimiento de patrones se limitaba a tareas específicas como el reconocimiento de caracteres en documentos impresos y la clasificación de datos en áreas como la biología y la estadística. No fue sino hasta la introducción de AlexNet en 2012, una red neuronal convolucional profunda, que se produjo un avance disruptivo en la clasificación de imágenes a gran escala. Este modelo demostró el potencial del *deep learning* para reconocer patrones complejos y superó significativamente los enfoques tradicionales, como quedó evidenciado en la

competencia ILSVRC-2012, donde redujo de manera considerable la tasa de error (Krizhevsky, Sutskever y Hinton, 2012).

Con los avances en la computación y el desarrollo de redes neuronales profundas, el reconocimiento de patrones ha ampliado su alcance a diversas disciplinas, incluyendo la visión por computadora, la robótica y la automatización industrial.

El impacto de AlexNet y el desarrollo de técnicas avanzadas en inteligencia artificial, han permitido la integración del reconocimiento de patrones en diversos entornos, incluyendo los sistemas embebidos. Estos sistemas, con recursos computacionales limitados, han comenzado a adoptar modelos optimizados de *deep learning* para mejorar la eficiencia en la toma de decisiones en tiempo real, sin depender de servidores externos.

Los sistemas embebidos se caracterizan por su diseño optimizado en términos de consumo energético, tamaño compacto y capacidades computacionales específicas, lo que facilita su integración en dispositivos con restricciones particulares de hardware (Novac et al., 2021). Estas plataformas tecnológicas surgieron para satisfacer la necesidad de ejecutar procesamiento en tiempo real en aplicaciones críticas donde no es viable depender de una conexión constante a servidores externos, como en sistemas industriales, automotrices, etc (Liu et al., 2020). Con el avance de la tecnología, los sistemas embebidos han evolucionado significativamente, permitiendo la implementación de técnicas avanzadas de reconocimiento de patrones para la toma de decisiones en tiempo real sin recurrir al procesamiento en la nube (Blanco-Filgueira et al., 2021). Esto ha conducido al desarrollo de aplicaciones más eficientes y autónomas en entornos con recursos escasos. Sin embargo, la implementación de modelos de aprendizaje profundo en hardware embebido continúa presentando desafíos en términos de eficiencia computacional, consumo energético y accesibilidad (Novac et al., 2021).

La identificación precisa de patrones en entornos de *Edge AI* es fundamental para diversas aplicaciones, como el control autónomo de robots, la detección de objetos en tiempo real y la optimización de sistemas inteligentes (Wan, Lele, & Raychowdhury, 2022). A pesar de los avances logrados, persiste la necesidad de desarrollar marcos metodológicos específicos que faciliten una mejor implementación y optimización de estos modelos en dispositivos con capacidades limitadas (Baller et al., 2021).

1.3 Definición y Descripción del Problema

El diseño e implementación de modelos de *deep learning* en sistemas embebidos presentan múltiples dificultades, tales como la reducción de la carga computacional, sin afectar la precisión del modelo, y la optimización de los recursos disponibles para maximizar su desempeño.

El mayor desafío de ejecutar algoritmos de inteligencia artificial, en tiempo real en el borde sobre sistemas embebidos de bajo costo, radica en las limitaciones de hardware y restricciones de recursos. Estos desafíos incluyen:

- Capacidad de procesamiento: Los sistemas embebidos de bajo costo suelen contar con procesadores de baja potencia, lo que dificulta la ejecución de modelos complejos de *deep learning* en tiempo real.
- Memoria limitada: La cantidad reducida de RAM y de almacenamiento, restringen la posibilidad de cargar y ejecutar modelos grandes de inteligencia artificial.
- Consumo energético: Muchos sistemas embebidos operan en entornos con restricciones de energía, por lo que la ejecución de modelos debe ser altamente eficiente.
- Latencia en la inferencia: Los modelos de IA requieren tiempos de respuesta rápidos para aplicaciones en tiempo real. En hardware limitado, reducir la latencia sin perder precisión es un desafío clave.
- Optimización de modelos: Es necesario reducir el tamaño de los modelos mediante técnicas como cuantización, poda de redes neuronales y modelos ligeros (MobileNet, TinyML) para adaptarlos a las capacidades del hardware embebido.
- Procesamiento en el borde: A diferencia de soluciones basadas en la nube donde hay acceso a recursos ilimitados, en el *edge computing* se requiere maximizar la eficiencia computacional en hardware con capacidades reducidas.
- Condiciones externas cambiantes: Los modelos deben ser robustos ante variaciones del entorno sin la posibilidad de realizar entrenamiento continuo, debido a la falta de conectividad constante.

Es de suma importancia iniciar con el desarrollo de marcos metodológicos, que faciliten la implementación de algoritmos de aprendizaje profundo en hardware embebido, equilibrando el rendimiento, la eficiencia y la accesibilidad.

1.4 Justificación

El presente trabajo de investigación, responde a la necesidad de desarrollar soluciones eficientes para la detección de patrones en sistemas embebidos de bajo costo, abordando desafíos clave en términos de capacidad de procesamiento, memoria y consumo energético. La relevancia de este estudio radica en su contribución a la optimización de modelos de inteligencia artificial para su ejecución en el borde, permitiendo su aplicación en entornos donde la conectividad con la nube no es viable o económicamente sostenible.

Desde una perspectiva práctica, este trabajo busca permitir la mejora en la toma de decisiones en tiempo real, dentro de sistemas con baja capacidad computacional, lo que resulta fundamental en áreas como la robótica autónoma, la automatización industrial y la vigilancia inteligente. Al minimizar la dependencia de servidores externos y mejorar la eficiencia computacional, la investigación busca contribuir en la reducción de costos operativos, mejorar la autonomía de dispositivos y optimizar la respuesta ante cambios en el entorno.

Además, este estudio introduce una metodología clara para la implementación de modelos de *deep learning* en hardware con recursos limitados, lo que representa un avance significativo en la democratización de la inteligencia artificial para aplicaciones en dispositivos de bajo costo.

La validación experimental, mediante el caso de uso del sumo bot, proporciona una base tangible para evaluar el desempeño del enfoque propuesto, sentando un precedente para futuras aplicaciones en distintos dominios.

1.5 Viabilidad

La viabilidad de este proyecto se fundamenta en la accesibilidad a los recursos técnicos y metodológicos necesarios para llevar a cabo la investigación.

1.5.1 Punto de Vista Técnico.

El estudio emplea hardware embebido de bajo costo, como Nicla Vision y Portenta H7 Vision Bundle, los cuales están ampliamente disponibles en el mercado y cuentan con soporte para *frameworks* de inteligencia artificial, optimizados para dispositivos con recursos limitados, tales como TensorFlow Lite y OpenMV. Además,

el desarrollo de la metodología propuesta se basa en técnicas establecidas en la literatura y en herramientas de código abierto, lo que garantiza su implementación sin restricciones de licencias propietarias.

Adicionalmente, el investigador tiene formación en *machine learning*, *deep learning*, procesamiento en el borde y *computer vision*, lo que permite abordar los desafíos inherentes a la ejecución de modelos en sistemas embebidos. En caso de ser necesario, se prevé la posibilidad de fortalecer conocimientos a través de cursos y literatura especializada en técnicas de optimización de modelos para hardware con recursos limitados. Esto garantiza que el investigador esté técnicamente capacitado para diseñar, implementar y validar la solución propuesta.

1.5.2 Punto de Vista Operativo.

Desde una perspectiva operativa, la validación del modelo se realiza en un entorno controlado, utilizando un dojo de 80 cm de diámetro y robots diseñados específicamente para la experimentación, asegurando la replicabilidad de los resultados. La experimentación se lleva a cabo en un laboratorio equipado con los recursos necesarios para la ejecución y evaluación del sistema, el cual está completamente a disposición del investigador sin incurrir en ningún costo adicional.

1.5.3 Punto de Vista Económico.

Desde el punto de vista económico, la realización de este proyecto es viable debido a que los costos asociados serán asumidos por el investigador. Los principales costos teóricos incluyen las horas de consultoría del investigador, la capacitación metodológica, adquisición de literatura y posibles licencias de software.

Este estudio no requiere una inversión significativa en infraestructura, ya que los dispositivos embebidos utilizados ya se encuentran disponibles y son de bajo costo en comparación con soluciones industriales. De igual forma, la investigación se lleva a cabo con los recursos computacionales y de laboratorio disponibles, minimizando los costos asociados a la experimentación. Adicionalmente, el uso de software de código abierto reduce significativamente los costos relacionados con licencias y plataformas comerciales.

Para lograr definir el monto por hora del investigador, es indispensable tomar en consideración diversos factores: la experiencia del profesional, la complejidad del proyecto y la ubicación geográfica.

De acuerdo a una recopilación de varias fuentes, como Clockify, en el 2024 (Tarifas horarias medias (2024): Freelancers y consultores), los consultores y *freelancers* en el área de TI/Software, presentan tarifas horarias medias entre $90 y $99.

Si se toma en cuenta la ubicación, Centroamérica presenta la tarifa horaria más baja: $18, siendo la tarifa global media: $23.

Al basarse en la experiencia laboral de un desarrollador de Software y siendo el mercado Norteamérica una referencia válida para la industria costarricense, los salarios por hora, se presentan de la siguiente manera:

- Menos de un año: $25,47
- Entre 1 y 4 años: $30,41
- Entre 5 y 9 años: $44,15
- Entre 10 y 19 años: $55,44
- Más de 20 años: $58,83

Para efectos de la investigación y basados en los datos anteriores, se decide utilizar un monto por hora de investigación de: $25.

Investigador	**Horas Semanales**	**Duración del proyecto en semanas**	**Duración total del proyecto en horas**	**Costo total del investigador en dólares**
Sergio Oviedo	25	13	325	$8,125

Tabla 3. Costo del investigador. Fuente: Elaboración propia

1.6 Objetivos

Para este proyecto, se utiliza la taxonomía de Bloom de 1956 ya que proporciona:

- Estructuración del contenido: Brinda una estructura jerárquica que ayuda a organizar el contenido del proyecto de manera lógica y coherente. Esto facilita la presentación de ideas de manera clara y ordenada.
- Definición de objetivos de aprendizaje: Ayuda a definir objetivos de aprendizaje claros y medibles para el proyecto. Esto asegura que el proyecto esté alineado con los objetivos educativos y que los estudiantes puedan demostrar el dominio de diferentes niveles de habilidades cognitivas.

- Existe una amplia documentación al respecto.

1.6.1 Objetivo General.

Desarrollar un marco metodológico, para la identificación de patrones en sistemas embebidos que trabajan en el borde en tiempo real, mediante el uso de algoritmos de aprendizaje profundo y *machine learning*, optimizando su aplicación en hardware de bajo costo y validando su desempeño en competiciones de sumo bot.

1.6.2 Objetivos Específicos.

1. Analizar las arquitecturas y enfoques actuales de procesamiento en el borde para la identificación de patrones en sistemas embebidos.
2. Seleccionar los algoritmos de *deep learning* y *machine learning* más adecuados para la detección de patrones en hardware de bajo costo.
3. Diseñar un marco metodológico que optimice la implementación de algoritmos de visión por computadora en dispositivos embebidos con recursos limitados.
4. Implementar el marco metodológico en dispositivos Nicla Vision y Portenta H7, evaluando su eficiencia computacional y consumo energético.
5. Validar el desempeño del marco metodológico en el caso de estudio de sumo bot, asegurando su precisión en la detección del dojo y los robots en tiempo real.

1.7 Alcances y Limitaciones

En este apartado, se busca definir y delimitar el marco de trabajo del proyecto, con el objetivo de proporcionar una visión clara de lo que se pretende lograr y las posibles restricciones o condiciones, que pueden influir en el desarrollo y los resultados del proyecto.

1.7.1 Alcances.

El alcance de este trabajo, comprende todos los aspectos descritos en el presente documento, enfocándose en la revisión y análisis de arquitecturas y algoritmos de inteligencia artificial, la validación experimental del marco metodológico y la evaluación de desempeño del modelo, en términos de precisión, eficiencia computacional y consumo energético de los dispositivos embebidos a utilizar.

1.7.2 Limitaciones.

Las siguientes son las limitaciones del presente trabajo:

1. El estudio está basado solo en el uso de los dispositivos Nicla Vision y Portenta H7 Vision Bundle, por lo que los resultados pueden variar en otro tipo de hardware.
2. La validación del trabajo, se realiza únicamente en el escenario específico de competiciones de sumo bot, por lo tanto si se busca aplicar en otros contextos industriales o académicos, puede requerir ajustes adicionales.
3. Debido a la naturaleza de los dispositivos utilizados, el procesamiento se debe optimizar para operar dentro de sus restricciones de memoria, potencia y capacidad de cómputo, lo que influye en la complejidad de los modelos de *machine learning* utilizados.
4. La detección de patrones en sumo bot depende de factores como iluminación, ángulos de cámara y contraste del dojo, lo que requiere ajustes adicionales al momento de capturar la imagen en tiempo real.

1.8 Revisión de literatura

En esta sección se presenta el proceso sistemático de búsqueda, recopilación, análisis y síntesis de información relevante y pertinente relacionada con el tema.

1.8.1 Revisión sistemática

Una revisión sistemática, es un proceso estructurado que permite recopilar, evaluar y sintetizar la evidencia disponible sobre un tema específico. En este trabajo, se sigue la metodología propuesta por Biolchini, Gomes, Cruz y Horta (2005). El objetivo de esta revisión es analizar la investigación existente en el ámbito de la optimización de modelos de IA para el reconocimiento de patrones en tiempo real, en sistemas embebidos de bajo costo para *edge computing* y lograr reunir y analizar la evidencia disponible sobre la efectividad, eficacia, eficiencia, impacto y/o calidad de las intervenciones o prácticas evaluadas.

1.8.1.1 Formulación de la pregunta

La formulación de la pregunta de investigación es un paso crucial en el proceso de revisión sistemática, ya que define el alcance y la dirección del estudio. Una pregunta bien estructurada permite identificar con precisión la información relevante en la literatura científica y facilita la síntesis de los hallazgos. Su importancia radica en garantizar que la búsqueda sea exhaustiva y enfocada, evitando sesgos y proporcionando una base sólida para el análisis de los datos recopilados.

1.8.1.1.1 Foco de la pregunta

Es el punto central o el objetivo principal que se busca explorar, analizar o responder mediante la investigación. Su función es dirigir la revisión hacia un área específica del conocimiento, asegurando que la recopilación de información sea relevante y efectiva. Un foco bien definido permite analizar la evolución del tema en la literatura, identificar los principales desafíos y detectar oportunidades futuras en el campo de estudio.

1.8.1.1.2 Amplitud y calidad de la pregunta

La amplitud de la pregunta, se relaciona con el grado de cobertura que tendrá el estudio en términos de alcance temático y profundidad analítica. Una pregunta amplia permite explorar diversas perspectivas y metodologías, mientras que una pregunta más específica se centra en un aspecto particular del tema. La amplitud debe ser equilibrada para garantizar que la revisión sea exhaustiva sin volverse demasiado general o dispersa.

1. Problema.

El problema real que se busca abordar en esta investigación es el desarrollo de estrategias eficientes para mejorar el rendimiento de algoritmos de reconocimiento de patrones en hardware embebido de bajo costo, permitiendo su implementación efectiva en el borde, garantizando tiempos de inferencia mínimos y alta eficiencia en entornos dinámicos. Los principales retos incluyen el diseño de modelos que maximicen el rendimiento en hardware con recursos limitados, la reducción de latencia sin comprometer la precisión, la optimización del consumo energético para aplicaciones en tiempo real y el desarrollo de técnicas que permitan una ejecución eficiente en entornos con restricciones computacionales. Además, estos sistemas deben ser capaces de operar en condiciones de iluminación variable, cambios en la distancia y otros factores ambientales fluctuantes que afectan la precisión y estabilidad del reconocimiento de patrones. Para superar estos desafíos, es fundamental desarrollar marcos metodológicos específicos que permitan equilibrar el rendimiento, la eficiencia energética y la accesibilidad de los algoritmos de inteligencia artificial.

2. Pregunta.

Basado en la definición del problema planteado anteriormente, se formula la siguiente pregunta de investigación:

¿Cómo optimizar los algoritmos de reconocimiento de patrones en hardware embebido de bajo costo, para mejorar el rendimiento y garantizar tiempos de

inferencia mínimos en entornos con procesamiento descentralizado y condiciones ambientales variables?

3. Palabras clave y sinónimos

Las palabras clave y expresiones relevantes derivadas de la pregunta de investigación, que pueden utilizarse en búsquedas bibliográficas, se presentan en la Tabla 4. Estas palabras han sido seleccionadas considerando su impacto en la precisión y exhaustividad de la búsqueda.

Palabra clave en idioma español	**Traducción o equivalente en el idioma inglés**
Optimización de algoritmos	Algorithm optimization
Reconocimiento de patrones	Pattern recognition
Hardware embebido	Embedded hardware
Rendimiento	Performance
Tiempo de inferencia	Inference time
Procesamiento descentralizado	Decentralized processing
Eficiencia computacional	Computational efficiency
Precisión del modelo	Model accuracy
Computación en el borde	Edge computing
IA en tiempo real	Real-time AI
Aprendizaje profundo	Deep learning
Hardware de bajo costo	Low-cost hardware

Tabla 4. Listado de palabras. Fuente: Elaboración propia

4. Intervención

Para mejorar el rendimiento de los algoritmos de reconocimiento de patrones en hardware embebido de bajo costo, se emplean diversas estrategias de optimización. Estas incluyen, técnicas avanzadas como la cuantización, poda de redes neuronales y compresión de modelos, con el objetivo de reducir el consumo de recursos sin comprometer la precisión. Además, se desarrollan enfoques que minimicen los tiempos de inferencia y maximicen la estabilidad del sistema en entornos con condiciones ambientales variables. La implementación de estas estrategias, busca garantizar que los sistemas embebidos operen de manera eficiente en tiempo real, adaptándose a cambios en iluminación, distancia y ruido sin depender del procesamiento en la nube.

5. Control

En este trabajo el control se define, como el conjunto de criterios y mecanismos utilizados para evaluar el desempeño de los algoritmos de reconocimiento de patrones, en un entorno de competencia de sumo bot. Dado que el *dataset* es propio y basado en un dojo con robots reales, el control se establece mediante métricas cuantificables, que permiten validar la efectividad de los modelos implementados. Los principales aspectos a evaluar son: la precisión del reconocimiento de los robots dentro del dojo, la estabilidad del algoritmo ante variaciones ambientales (iluminación y distancia) y la capacidad del sistema para operar en tiempo real con tiempos de inferencia mínimos. Para ello, se diseñan pruebas experimentales que simulan escenarios de competencia reales, garantizando la fiabilidad y reproducibilidad de los resultados.

6. Efectos

Los efectos de esta investigación, se centran en evaluar y optimizar el rendimiento de los algoritmos de reconocimiento de patrones en hardware embebido de bajo costo, asegurando su funcionamiento eficiente en entornos dinámicos. A través de la revisión sistemática, se analizan estrategias existentes y sus limitaciones, para identificar mejoras que reduzcan los tiempos de inferencia, optimicen el uso de recursos computacionales y aumenten la estabilidad en la detección de patrones bajo condiciones ambientales cambiantes. Además, la aplicación de técnicas de optimización, tienen como objetivo incrementar la precisión y confiabilidad del sistema, validando su efectividad en escenarios prácticos, como la competencia de sumo bot.

7. Medida de salida

Las medidas de salida en esta investigación, se definen mediante indicadores cuantificables, que permiten evaluar la efectividad de las estrategias de optimización aplicadas a los algoritmos de reconocimiento de patrones. Se busca garantizar un equilibrio entre rendimiento, eficiencia y adaptabilidad en entornos reales.

8. Población

La población de esta investigación, está conformada por imágenes y datos extraídos de escenarios simulados de competiciones de sumo bot. Mediante el análisis de estas imágenes, se podrá evaluar la efectividad de los algoritmos optimizados en condiciones reales de uso.

9. Aplicación

Los resultados de este estudio, pueden trascender a las competiciones de sumo bot, aplicándose en entornos que requieran procesamiento descentralizado y

eficiente en hardware con recursos limitados. Entre las aplicaciones potenciales, se incluyen sistemas autónomos, robótica industrial, automatización de procesos y soluciones de visión artificial dentro del Internet de las Cosas (IoT), donde la optimización del rendimiento y la eficiencia computacional son fundamentales para la ejecución en tiempo real.

10. Ciencia de Diseño

El enfoque de la ciencia del diseño, se basa en la creación y validación de estrategias para optimizar algoritmos de reconocimiento de patrones en hardware embebido de bajo costo. A través de un proceso iterativo, se desarrollan y prueban soluciones que mejoran el rendimiento, reducen los tiempos de inferencia y garantizan estabilidad en entornos con condiciones ambientales variables. La validación se lleva a cabo en un entorno experimental basado en la competencia de sumo bot, utilizando un *dataset* propio generado con robots reales en un dojo real, lo que permite evaluar el impacto de las optimizaciones en un contexto práctico y dinámico.

1.8.1.2 Selección de fuentes

En esta sección, se detallan las fuentes utilizadas para la identificación de estudios primarios en la investigación. Se ha tomado como referencia la plantilla Biolchini para estructurar el proceso.

1.8.1.2.1 Definición del criterio de selección de fuentes

Para garantizar la validez de las fuentes, se han considerado las siguientes métricas:

- H-Index mayor a 20.
- Quartiles: Q1.
- SJR mayor a 1.
- Citaciones por documento mayor a 4.

1.8.1.2.2 Lenguaje de estudio

Dado que este estudio se desarrolla en Costa Rica, el español es el idioma principal del trabajo. Sin embargo, debido al alcance global de la investigación y la abundancia de información en inglés, este idioma también es utilizado con frecuencia para garantizar el acceso a fuentes relevantes y la estandarización de términos técnicos.

1.8.1.2.3 Identificación de fuentes

1. Método de selección de fuentes:

   Se emplea Google Académico para la búsqueda de referencias.

2. Cadena de búsqueda principales:

- "pattern recognition" AND "embedded systems" AND "low-cost hardware" AND "optimization"
- "real-time AI" AND "edge computing" AND "efficient inference" AND "embedded devices"
- "deep learning optimization" AND "quantization" AND "neural network pruning" AND "low-power AI"
- "detección de objetos" AND "procesamiento eficiente" AND "visión artificial"

3. Lista de fuentes seleccionadas:
    1. IEEE Xplore
    2. IEEE Access
    3. Journal of King Saud University
    4. ACM Digital Library

1.8.1.2.4 Selección de fuentes después de la evaluación

Tras aplicar los filtros de calidad, se seleccionaron los estudios que ofrecían el contenido más relevante y garantizaban acceso completo a su información.

1.8.1.2.5 Comprobación de las fuentes

Al momento de realizar esta sección del proyecto, no se contaba con una mesa de expertos para analizar los resultados de las búsquedas escogidas.

1.8.1.3 Selección de los estudios

Esta sección establece el método para seleccionar estudios.

1.8.1.3.1 Definición del criterio de inclusión y exclusión de estudios

Los estudios incluidos deben cumplir los siguientes requisitos:

- Publicados después de 2021.
- Aplicar algoritmos de ML y/o *deep learning*.
- Enfocados en hardware embebido, de bajo costo y con capacidad de operar en el borde.
- Presentar un enfoque cuantitativo en sus resultados.

1.8.1.3.2 Definición de tipos de estudio

Se seleccionan estudios con un enfoque metodológico sólido y alta relevancia en la optimización de algoritmos de reconocimiento de objetos, considerando sus condiciones y limitaciones en el momento de la implementación. También se incluyen aquellos que presenten métricas cuantificables y aplicaciones en escenarios reales.

1.8.1.3.3 Procedimiento para la selección de los estudios

La selección de estudios, se realiza mediante la lectura completa del abstract y el análisis del documento con la técnica de “skimming”. Se priorizarán aquellos que aborden desafíos como precisión, tiempos de inferencia y consumo energético, y que presenten resultados cuantificables.

4.8.1.4 Tabla de los resultados de la revisión literaria

4.8.1.4.1 Fuente:

Título	AIfES: A Next-Generation Edge AI Framework
Publicación	IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 46, NO. 6, JUNE 2024
Autor(es)	Lars Wulfert, Johannes Kühnel, Lukas Krupp, Justus Viga, Christian Wiede, Pierre Gembaczka
Referencia	L. Wulfert et al., "AIfES: A Next-Generation Edge AI Framework," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 46, no. 6, pp. 4519-4533, June 2024, doi: 10.1109/TPAMI.2024.3355495. keywords: {Training;Data models;Artificial intelligence;Support vector machines;Hardware acceleration;Libraries;Performance evaluation;Edge AI Framework;embedded systems;machine learning framework;on-device training;resource-constrained devices;TinyML},
Área	Inteligencia Artificial
Resumen	El artículo presenta AIfES (Artificial Intelligence for Embedded Systems), un nuevo marco de inteligencia artificial para sistemas embebidos. Este *framework* ofrece una solución flexible, modular y eficiente para la ejecución y entrenamiento de modelos de *machine learning* en dispositivos con recursos limitados, superando las restricciones de las soluciones tradicionales de Edge AI. Comparado con TensorFlow Lite for Microcontrollers (TFLM) en un ARM Cortex-M4, AIfES reduce el consumo de memoria hasta en un 54% y mejora el rendimiento en redes neuronales totalmente conectadas (FCNNs) y convolucionales (CNNs). Además, permite el entrenamiento en el propio dispositivo, facilitando su uso en aprendizaje federado y aprendizaje continuo en entornos con capacidad computacional reducida.
Aspectos a destacar	

El artículo presenta un nuevo marco de inteligencia artificial para sistemas embebidos, destacando su capacidad de entrenamiento en el dispositivo para facilitar el aprendizaje continuo sin depender de la nube. Su arquitectura modular permite integrar aceleradores de hardware personalizados, optimizando el procesamiento en dispositivos con recursos limitados. Su naturaleza de código abierto y compatibilidad con RISC-V, ARM Cortex-M y ESP32 facilitan su adopción y personalización para diversos usos en Edge AI.

Tabla 5.IEEE Transactions on Pattern Analysis and Machine Intelligence, 2024. Fuente: elaboración propia

4.8.1.4.2 Fuente:

Título	Anatomy of Deep Learning Image Classification and Object Detection on Commercial Edge Devices: A Case Study on Face Mask Detection
Publicación	IEEE Access, 2022, Volume 10
Autor(es)	Dimitrios Kolosov, Vasilios Kelefouras, Pandelis Kourtessis, Iosif Mporas
Referencia	D. Kolosov, V. Kelefouras, P. Kourtessis and I. Mporas, "Anatomy of Deep Learning Image Classification and Object Detection on Commercial Edge Devices: A Case Study on Face Mask Detection," in IEEE Access, vol. 10, pp. 109167-109186, 2022, doi: 10.1109/ACCESS.2022.3214214. keywords: {Integrated circuit modeling;Optimization;Hardware;Computational modeling;Integrated circuits;Image edge detection;Face recognition;Computer vision;Performance evaluation;Edge computing;Object detection;Image classification;Image classification;object detection;edge computing;computer vision;performance evaluation},
Área	Inteligencia Artificial
Resumen	El artículo analiza la implementación de modelos de *deep learning* en Edge AI, comparando su precisión y eficiencia en la detección de mascarillas faciales en hardware de bajo costo. Examina los desafíos de ejecutar redes neuronales profundas en dispositivos con recursos limitados y explora estrategias de optimización como la cuantización y la poda de redes.
Aspectos a destacar	
El artículo demuestra que la optimización de modelos de *deep learning* permite la detección de mascarillas en tiempo real en Edge AI, pese a las limitaciones de hardware. Destaca la importancia de reducir el consumo de memoria y mejorar la velocidad de inferencia mediante cuantización y poda de redes neuronales, asegurando un equilibrio entre precisión y eficiencia. Los resultados validan la viabilidad de Edge AI en aplicaciones de seguridad y salud pública, proporcionando una referencia clave para soluciones de visión artificial en entornos con recursos limitados.	

Tabla 6. Anatomy of Deep Learning Image Classification and Object Detection on Commercial Edge Devices: A Case Study on Face Mask Detection. Fuente: elaboración propia

4.8.1.4.3 Fuente:

Título	Efficient Acceleration of Deep Learning Inference on Resource-Constrained Edge Devices: A Review

Publicación	Proceedings of the IEEE, Volume 111, Issue 1
Autor(es)	Md. Maruf Hossain Shuvo, Syed Kamrul Islam, Jianlin Cheng, Bashir I. Morshed
Referencia	M. M. H. Shuvo, S. K. Islam, J. Cheng and B. I. Morshed, "Efficient Acceleration of Deep Learning Inference on Resource-Constrained Edge Devices: A Review," in Proceedings of the IEEE, vol. 111, no. 1, pp. 42-91, Jan. 2023, doi: 10.1109/JPROC.2022.3226481. keywords: {Edge computing;Image edge detection;Real-time systems;Cloud computing;Artificial intelligence;Optimization;Computer architecture;Neural networks;Deep learning;Algorithm–hardware codesign;artificial intelligence (AI);artificial intelligence on edge (edge-AI);deep learning (DL);model compression;neural accelerator},
Área	Inteligencia Artificial, IoT
Resumen	El artículo revisa técnicas avanzadas para mejorar la inferencia de *deep learning* en Edge AI, abordando los desafíos de implementar redes neuronales en hardware con recursos limitados. Destaca la importancia de la optimización en hardware y software para maximizar el rendimiento. Identifica cuatro áreas clave de investigación: diseño de arquitecturas, optimización de métodos, codesarrollo de hardware y algoritmos, y aceleradores eficientes. Presenta estrategias como cuantización, poda de redes y modelos ligeros, permitiendo ejecutar IA en dispositivos embebidos e IoT con eficiencia y precisión.
Aspectos a destacar	
El artículo destaca la optimización de *deep learning* en Edge AI, enfocándose en cuantización, poda de redes y modelos ligeros para mejorar la eficiencia sin perder precisión. Resalta el codesarrollo de hardware y software, incluyendo aceleradores especializados para reducir el consumo energético y los tiempos de inferencia. Estas estrategias son clave para aplicaciones en IoT**,** automatización y sistemas autónomos.	

Tabla 7. Efficient Acceleration of Deep Learning Inference on Resource-Constrained Edge Devices: A Review. Fuente: elaboración propia

2.8.1.4.4 Fuente:

Título	A review on TinyML: State-of-the-art and prospects
Publicación	Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 4,2022, Pages 1595-1623, ISSN 1319-1578
Autor(es)	Partha Pratim Ray
Referencia	Ray, P. P. (2021). A review on TinyML: State-of-the-art and prospects. Journal Of King Saud University - Computer And Information Sciences, 34(4), 1595-1623. https://doi.org/10.1016/j.jksuci.2021.11.019
Área	Inteligencia Artificial
Resumen	
Aspectos a destacar	

Tabla 8. A review on TinyML: State-of-the-art and prospects. Fuente: elaboración propia

2.8.1.4.5 Fuente:

Título	Collaborative Inference in Resource-Constrained Edge Networks: Challenges and Opportunities
Publicación	MILCOM 2024 - 2024 IEEE Military Communications Conference (MILCOM)
Autor(es)	Nathan Ng, Abel Souza, Suhas Diggavi, Niranjan Suri, Tarek Abdelzaher, Don Towsley
Referencia	N. Ng et al., "Collaborative Inference in Resource-Constrained Edge Networks: Challenges and Opportunities," MILCOM 2024 - 2024 IEEE Military Communications Conference (MILCOM), Washington, DC, USA, 2024, pp. 1-6, doi: 10.1109/MILCOM61039.2024.10773876. keywords: {Performance evaluation;Military communication;Thermal factors;Image edge detection;Collaboration;Machine learning;Servers;Resource management;Internet of Things;Hardware acceleration;Collaborative inference;edge computing},
Área	Inteligencia Artificial, IoT, Sistemas Embebidos
Resumen	El artículo, proporciona un análisis exhaustivo del paradigma TinyML, que busca ejecutar modelos de aprendizaje automático en dispositivos embebidos con recursos limitados. Se exploran sus fundamentos, herramientas y desafíos, destacando cómo la computación en el borde y el Internet de las Cosas (IoT) han impulsado la necesidad de modelos más eficientes en términos de consumo energético y memoria. Además, se describen enfoques clave como la cuantización, la búsqueda de arquitectura neuronal optimizada y la compresión de modelos, los cuales permiten que el aprendizaje profundo pueda ejecutarse en hardware con restricciones computacionales.
Aspectos a destacar	
El artículo destaca la importancia de TinyML en la ejecución de modelos de aprendizaje automático en dispositivos embebidos, optimizando Edge AI e IoT. Enfatiza técnicas como cuantización, poda de redes y compresión de modelos para mejorar la eficiencia sin perder precisión. Se presentan aplicaciones en reconocimiento de imágenes, detección de voz y gestos, vehículos autónomos y monitoreo en tiempo real, subrayando desafíos en estandarización, evaluación de modelos y adaptabilidad a diversas arquitecturas de hardware.	

Tabla 9. Collaborative Inference in Resource-Constrained Edge Networks: Challenges and Opportunities. Fuente: elaboración propia

2.8.1.4.6 Fuente:

Título	Human Activity Recognition on Microcontrollers with Quantized and Adaptive Deep Neural Networks
Publicación	ACM Transactions on Embedded Computing Systems (TECS), Volume 21, Issue 4Article No.: 46, Pages 1–28https://doi.org/10.1145/3542819

Autor(es)	Francesco Daghero, Alessio Burrello, Chen Xie, Marco Castellano, Luca Gandolfi, Andrea Calimera, Enrico Macii, Massimo Poncino, Daniele Jahier Pagliari
Referencia	Daghero, F., Burrello, A., Xie, C., Castellano, M., Gandolfi, L., Calimera, A., ... & Pagliari, D. J. (2022). Human activity recognition on microcontrollers with quantized and adaptive deep neural networks. *ACM Transactions on Embedded Computing Systems (TECS)*, *21*(4), 1-28.
Área	Inteligencia Artificial
Resumen	El artículo, presenta un enfoque eficiente para el reconocimiento de actividades humanas (HAR) en microcontroladores de bajo consumo, utilizando redes neuronales convolucionales (CNNs) cuantizadas y adaptativas. Debido a las limitaciones computacionales de estos dispositivos, los métodos tradicionales de aprendizaje profundo no son viables, por lo que los autores proponen optimizaciones basadas en sub-byte y precisión mixta, logrando un equilibrio entre precisión y uso de memoria. Además, se introduce la inferencia adaptativa, que ajusta la complejidad de la inferencia en tiempo de ejecución según la dificultad de la entrada, mejorando la flexibilidad del sistema sin un impacto significativo en el consumo de memoria.
Aspectos a destacar	
El artículo destaca la implementación de redes neuronales convolucionales cuantizadas y adaptativas para el reconocimiento de actividades humanas (HAR) en microcontroladores de bajo consumo, optimizando el uso de memoria y el tiempo de inferencia. Se introduce la inferencia adaptativa, que ajusta la complejidad del modelo en tiempo de ejecución, logrando una reducción del uso de memoria de más de 100 veces y tiempos de inferencia de entre 9 µs y 16 ms en un microcontrolador RISC-V de ultra bajo consumo. Los experimentos demuestran que el enfoque propuesto supera a los métodos tradicionales en tres de cuatro conjuntos de datos, confirmando su viabilidad para aplicaciones embebidas con restricciones energéticas.	

Tabla 10. Human Activity Recognition on Microcontrollers with Quantized and Adaptive Deep Neural Networks Fuente: elaboración propia

2.8.1.4.7 Fuente:

Título	Train++: An Incremental ML Model Training Algorithm to Create Self-Learning IoT Devices
Publicación	2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI)
Autor(es)	Bharath Sudharsan, Piyush Yadav, John G. Breslin, Muhammad Intizar Ali
Referencia	B. Sudharsan, P. Yadav, J. G. Breslin and M. Intizar Ali, "Train++: An Incremental ML Model Training Algorithm to Create Self-Learning IoT Devices," 2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation

	(SmartWorld/SCALCOM/UIC/ATC/IOP/SCI), Atlanta, GA, USA, 2021, pp. 97-106, doi: 10.1109/SWC50871.2021.00023. keywords: {Training;Performance evaluation;Solid modeling;Embedded systems;Data models;Real-time systems;Inference algorithms;Intelligent Microcontrollers;Online Learning;Optimization;Incremental Learning;Edge Computing},
Área	Inteligencia Artificial
Resumen	El artículo presenta Train++, un algoritmo de entrenamiento incremental de modelos de aprendizaje automático diseñado para dispositivos IoT con recursos limitados. A diferencia de los enfoques tradicionales que requieren modelos entrenados en la nube, Train++ permite que los dispositivos IoT entrenen localmente con datos en vivo, lo que les permite adaptarse a cambios en los patrones de datos y operar sin dependencia de la nube.
Aspectos a destacar	
Train++ permite a dispositivos IoT con recursos limitados, entrenar modelos de *machine learning* localmente, sin depender de la nube. Destaca por su capacidad de adaptarse a cambios en los datos en tiempo real, su bajo consumo de memoria y energía (hasta 65,000 veces menos que otros métodos), y por mejorar la precisión del modelo hasta en un 7.3%. Evaluado en múltiples microcontroladores y conjuntos de datos, logra entrenamientos más rápidos y eficientes, transformando dispositivos IoT en sistemas autónomos y autoaprendices.	

Tabla 11. Train++: An Incremental ML Model TrainingAlgorithm to Create Self-Learning IoT Devices. Fuente elaboración propia

2.8.1.4.8 Fuente:

Título	ML-MCU: A Framework to Train ML Classifiers on MCU-based IoT Edge Devices
Publicación	IEEE Internet of Things Journal, Volume 9, Issue 16
Autor(es)	Bharath Sudharsan, John G. Breslin, Muhammad Intizar Ali
Referencia	B. Sudharsan, J. G. Breslin and M. I. Ali, "ML-MCU: A Framework to Train ML Classifiers on MCU-Based IoT Edge Devices," in IEEE Internet of Things Journal, vol. 9, no. 16, pp. 15007-15017, 15 Aug.15, 2022, doi: 10.1109/JIOT.2021.3098166. keywords: {Training;Performance evaluation;Data models;HVAC;Stochastic processes;Internet of Things;Optimized production technology;Edge intelligence;intelligent microcontrollers;real-time machine learning (ML);self-learning IoT devices},
Área	Inteligencia Artificial
Resumen	El artículo ML-MCU, introduce un marco de aprendizaje automático diseñado para entrenar modelos en microcontroladores (MCUs) de dispositivos IoT, permitiéndoles autoaprendizaje sin depender de la nube. A través de los algoritmos Opt-SGD para clasificación binaria y Opt-OVO para clasificación multiclase, el sistema logra entrenar modelos eficientemente en hardware con memoria y

	procesamiento limitados. Evaluado en múltiples MCUs con distintos conjuntos de datos, ML-MCU demuestra la capacidad de entrenar modelos en dispositivos con pocos recursos, mejorar la precisión de las predicciones y realizar inferencias en tiempo real, transformando los dispositivos IoT en sistemas inteligentes y autónomos.
Aspectos a destacar	
Entre los aspectos más destacados de ML-MCU se encuentran su capacidad de autoaprendizaje en el borde, eliminando la necesidad de procesar datos en la nube, y sus algoritmos optimizados que permiten entrenar modelos de manera incremental sin sobrecargar la memoria. Además, su compatibilidad con hardware de bajos recursos lo hace viable para una amplia gama de dispositivos IoT, permitiendo inferencias en tiempo real en milisegundos. Finalmente, su enfoque en eficiencia energética y reducción del uso de memoria optimiza el desempeño de los dispositivos sin comprometer la precisión del modelo.	

Tabla 12. ML-MCU: A Framework to Train ML Classifierson MCU-based IoT Edge Devices. Fuente: elaboración propia

1.8.2 Estado de la cuestión

El desarrollo de algoritmos de inteligencia artificial en sistemas embebidos de bajo costo, ha cobrado gran relevancia en los últimos años, impulsado por la expansión del Internet de las Cosas (IoT), las limitaciones de la computación en la nube en aplicaciones de tiempo real y el avance de microcontroladores más potentes con bajo consumo energético. Estos avances, han permitido la ejecución de modelos en hardware con recursos limitados, optimizando su rendimiento sin comprometer la eficiencia.

A pesar del progreso en software y hardware, estos sistemas continúan enfrentando desafíos debido a restricciones en memoria, capacidad de procesamiento y consumo energético. Esto ha motivado el desarrollo de técnicas de optimización, para mejorar la eficiencia y precisión de los modelos de inteligencia artificial, logrando tiempos de inferencia mínimos. Entre las estrategias más relevantes se encuentran la cuantización y la poda de redes neuronales, que permiten reducir el tamaño de los modelos sin afectar significativamente su precisión. Asimismo, el uso de arquitecturas modulares y adaptativas optimiza la inferencia en microcontroladores, ajustando la ejecución del modelo según la complejidad de la entrada.

Otro aspecto clave, es el codesarrollo de hardware y software, donde la integración de aceleradores especializados y arquitecturas optimizadas, ha mejorado la eficiencia del aprendizaje profundo en dispositivos que poseen capacidades computacionales limitadas. Sin embargo, persisten desafíos como la falta de

estandarización en la evaluación y comparación de modelos en hardware embebido, la necesidad de mejorar la robustez ante condiciones ambientales variables y la exploración de nuevos enfoques que logren un equilibrio entre rendimiento, consumo energético y precisión en inferencia en tiempo real.

Capítulo 2. Marco Conceptual

A continuación, se muestra un mapa conceptual basado en una nube de conceptos que permite organizar y visualizar las ideas, términos o conceptos relacionados o contenidos en la sección “Estado de la cuestión”.

2.1 Nube de conceptos

Figura 3.Nube de palabras. Fuente: Elaboración propia. Elaborado usando el sitio: https://www.nubedepalabras.es

2.2 Mapa conceptual jerárquico

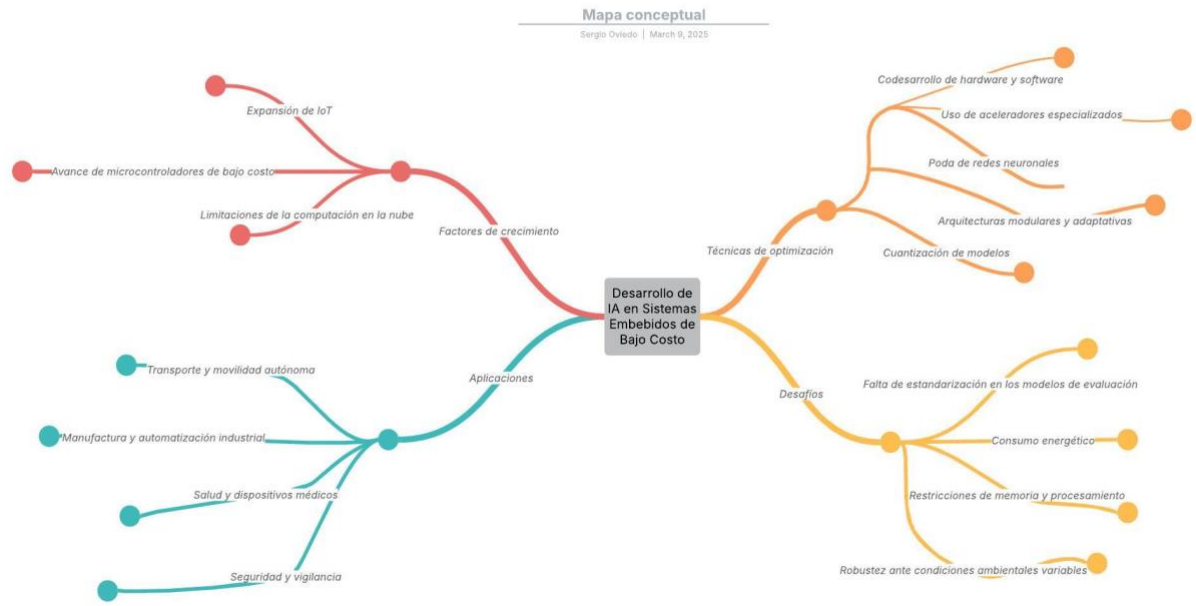

Figura 4. Mapa Conceptual. Elaboración propia. Elaborado usando el sitio: https://lucid.app

2.3 Definición de conceptos

En este apartado, se establece con precisión y claridad el significado y las características de los términos, ideas o conceptos clave que serán utilizados y explorados en el transcurso del proyecto.

2.3.1 Sistemas embebidos

De acuerdo al libro "*Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers*", su autora, Tammy Noergaard, describe el concepto como aquellos sistemas computacionales que se encuentran integrados en dispositivos más grandes, diseñados para realizar funciones específicas. Estos sistemas suelen tener restricciones de recursos y pueden operar en tiempo real, diferenciándose de las computadoras de propósito general.

2.3.2 Computación en el borde (*Edge Computing*)

La computación en el borde, es un paradigma que acerca el procesamiento de datos y los servicios de almacenamiento al lugar donde se generan los datos, es decir, en el "borde" de la red, con el objetivo de mejorar los tiempos de respuesta y ahorrar ancho de banda. (Chao, Wang, Wii, Li,Colman, Tornarle y Zhang, 2019)

2.3.3 Aprendizaje automático (*Machine Learning*)

De acuerdo a los autores de libro "*Deep Learning*" (Goodfellow, I., Bengio, Y., & Courville, A, 2016), el aprendizaje automático es una disciplina de la inteligencia artificial que se centra en el desarrollo de algoritmos que permiten a las computadoras aprender y mejorar a partir de la experiencia, sin ser programadas explícitamente para cada tarea. Estos algoritmos analizan datos y reconocen patrones para realizar predicciones o tomar decisiones basadas en la información proporcionada.

2.3.4 Aprendizaje profundo (*Deep Learning*)

El aprendizaje profundo es una subdisciplina del aprendizaje automático que se centra en el uso de redes neuronales con múltiples capas para modelar datos complejos. Estas redes permiten a las máquinas aprender representaciones de datos con múltiples niveles de abstracción, lo que mejora tareas como el reconocimiento de voz, la visión por computadora y el procesamiento del lenguaje natural (Goodfellow, I., Bengio, Y., & Courville, A, 2016).

2.3.5 Algoritmo IA (Inteligencia Artificial)

Es un conjunto de instrucciones diseñadas para que una máquina realice tareas que, si fueran realizadas por humanos, requerirían inteligencia. Estos algoritmos permiten a las máquinas procesar información, aprender de los datos y tomar decisiones basadas en patrones reconocidos (Russell, S., & Norvig, P., 2010).

Se utilizan en diversas aplicaciones, desde el reconocimiento de voz hasta la conducción autónoma, destacando su capacidad para mejorar y adaptarse con el tiempo, a medida que procesan más datos (Sharma, A., & Kaur, R.,2013).

2.3.6 Optimización de modelos de IA

Esta tarea se refiere al proceso de ajustar y mejorar los parámetros y estructuras de los modelos de IA para maximizar su rendimiento y eficiencia en tareas específicas. Este proceso es esencial para garantizar que los modelos de IA sean precisos y eficientes en términos de recursos computacionales y tiempo de procesamiento (Li, Y., & Shami, A., 2020).

2.3.7 Cuantización de modelos

La cuantización de modelos en inteligencia artificial es una técnica que reduce la precisión numérica de los pesos y activaciones de una red neuronal, transformándolos de formatos de alta precisión, como números de coma flotante de 32 bits (FP32), a formatos de menor precisión, como enteros de 8 bits (INT8). Este

proceso disminuye el tamaño del modelo y mejora la eficiencia computacional, permitiendo implementaciones más rápidas y con menor consumo de recursos (Nagel, M., Fournarakis, M., Amjad, R. A., Bondarenko, Y., van Baalen, M., & Blankevoort, T., 2021).

2.3.8 Poda de redes neuronales

La poda de redes neuronales es una técnica que busca mejorar la eficiencia de los modelos de inteligencia artificial al eliminar conexiones o neuronas que aportan poco o nada al rendimiento general de la red. Este proceso reduce la complejidad del modelo, disminuye el consumo de recursos computacionales y puede mejorar la capacidad de generalización al minimizar el sobreajuste (Hoefler, T., Alistarh, D., Ben-Nun, T., Dryden, N., & Peste, A. 2021).

2.3.9 Aceleradores de hardware

Los aceleradores de hardware son dispositivos especializados diseñados para mejorar el rendimiento y la eficiencia de los algoritmos de inteligencia artificial (IA) en sistemas embebidos que operan en el borde de la red. Estos dispositivos ayudan a incrementar el soporte a los modelos de IA y algoritmos (Liu et al., 2023).Además, contribuyen a una mayor eficiencia energética, lo cual es crucial en dispositivos con recursos limitados.

2.3.10 Arquitecturas modulares

Las arquitecturas modulares permiten adaptar sistemas embebidos de borde a diferentes requerimientos de procesamiento, favoreciendo la escalabilidad y mantenimiento del sistema (Han et al., 2020). Este enfoque facilita la integración de módulos de inteligencia artificial especializados, como motores de inferencia, gestores de datos o controladores de dispositivos, que pueden operar de forma autónoma o cooperativa según la carga de trabajo. En entornos con restricciones de energía y memoria, como los dispositivos edge, esta modularidad permite asignar dinámicamente los recursos, optimizando el rendimiento y reduciendo el consumo energético.

2.3.11 Arquitecturas adaptativas

Las arquitecturas adaptativas en sistemas embebidos de borde, se refieren a diseños que permiten que el sistema ajuste dinámicamente su comportamiento y recursos en función de las condiciones operativas y las cargas de trabajo, optimizando así el rendimiento y la eficiencia energética. Este enfoque es

fundamental para el desarrollo efectivo de sistemas de inteligencia artificial en el borde, tal como se discute en metodologías de co-diseño y especialización (Hao et al., 2021).

2.3.12 Eficiencia energética en IA

La eficiencia energética en inteligencia artificial (IA) aplicada a sistemas embebidos en el borde se refiere a la capacidad de estos sistemas para ejecutar modelos de IA minimizando el consumo de energía, sin comprometer significativamente el rendimiento ni la precisión (Feronato et al., 2023).

Capítulo 3. Marco Metodológico

3.1 Tipo de Investigación

La presente investigación se clasifica como aplicada, con enfoque cuantitativo, alcance explicativo y diseño experimental, ya que busca generar conocimiento útil y transferible para resolver un problema técnico específico: la optimización de algoritmos de reconocimiento de patrones basados en inteligencia artificial para su implementación eficiente en sistemas embebidos de bajo costo que operan en el borde (edge computing).

De acuerdo con Hernández Sampieri, Fernández y Baptista (2014), la investigación aplicada tiene como finalidad resolver problemas prácticos mediante el uso del conocimiento científico, a diferencia de la investigación básica, que se orienta principalmente al desarrollo teórico. En este caso, la optimización de modelos de visión por computadora (mediante técnicas como cuantización, reducción de parámetros o transformación de formatos) y su validación experimental en dispositivos con recursos computacionales limitados, representa una solución concreta a un desafío tecnológico actual en áreas como la robótica, IoT y la automatización embebida.

Asimismo, el diseño es de tipo experimental, ya que se manipulará intencionalmente una variable independiente —las técnicas de optimización aplicadas a un modelo base— con el fin de observar su efecto sobre variables dependientes como la latencia de inferencia, precisión, velocidad de procesamiento (FPS), uso de memoria y consumo energético, al ejecutar dichos modelos en plataformas embebidas específicas (Nicla Vision, Portenta H7 Vision Bundle). Estas pruebas se llevarán a cabo bajo condiciones controladas, que incluyen el uso de un *dataset* predefinido, una arquitectura de red constante, y un entorno físico replicable.

Según Hernández Sampieri et al. (2014), un experimento implica la manipulación intencionada de una o más variables independientes para observar su efecto sobre variables dependientes, en condiciones controladas que permitan establecer relaciones causales. Por lo tanto, el enfoque experimental adoptado en esta investigación es pertinente para explicar cómo diferentes estrategias de optimización afectan el rendimiento de modelos de IA en dispositivos embebido**s**, y validar empíricamente un marco metodológico replicable, mediante un caso de aplicación concreto: la detección de límites y robots en un entorno de competencia tipo sumo bot.

3.2 Alcance Investigativo

La presente investigación se enmarca dentro del alcance explicativo, ya que tiene como objetivo principal identificar, analizar y comprender las causas que influyen en el rendimiento de modelos de reconocimiento de patrones basados en inteligencia artificial, cuando son optimizados e implementados en hardware embebido de bajo costo. A diferencia de estudios descriptivos o correlacionales, este trabajo busca establecer relaciones de causa-efecto entre las técnicas de optimización aplicadas a los modelos y los resultados obtenidos en métricas clave, tales como la latencia de

inferencia, la precisión, el uso de memoria, la velocidad de procesamiento (FPS) y el comportamiento ante condiciones ambientales variables.

De acuerdo con Hernández Sampieri, Fernández y Baptista (2014), los estudios explicativos tienen como finalidad comprender las razones por las cuales ocurre un fenómeno y bajo qué condiciones se presenta, lo cual resulta pertinente en este caso, ya que se pretende explicar cómo ciertas estrategias de optimización afectan el desempeño de los modelos de IA en entornos de procesamiento descentralizado (edge computing).

Para ello, la investigación se apoya en un diseño experimental que permite manipular intencionadamente la variable independiente —es decir, la técnica de optimización empleada sobre un modelo base— con el fin de observar sus efectos sobre variables dependientes previamente definidas. Esta estructura permite analizar de forma empírica y controlada las condiciones bajo las cuales una optimización específica mejora (o no) el rendimiento del sistema, contribuyendo así al desarrollo de un marco metodológico aplicable a otros contextos donde se requiera ejecutar IA eficiente en el borde.

Este tipo de alcance es especialmente pertinente en investigaciones de carácter aplicado e ingenieril, ya que permite no solo medir el rendimiento, sino también explicar las causas que lo determinan, generando conocimiento útil y replicable para futuros desarrollos en inteligencia artificial embebida.

3.3 Enfoque

La presente investigación adopta un enfoque cuantitativo, ya que se centra en la recolección, análisis e interpretación de datos numéricos con el propósito de evaluar el impacto de distintas técnicas de optimización aplicadas a modelos de reconocimiento de patrones mediante inteligencia artificial, implementados en sistemas embebidos de

bajo costo. El estudio tiene como objetivo medir con precisión variables como la latencia de inferencia, la precisión del modelo, la velocidad de procesamiento (FPS), el uso de memoria y otros indicadores de rendimiento técnico, bajo condiciones controladas y replicables.

A través del uso de un diseño experimental, se manipulará la variable independiente —la técnica de optimización aplicada— para observar su efecto sobre las variables dependientes, utilizando herramientas especializadas de medición, benchmarking y análisis estadístico básico. De este modo, se busca establecer relaciones causales cuantificables que permitan validar empíricamente la eficacia de cada estrategia de optimización dentro del marco metodológico propuesto.

Según Hernández Sampieri, Fernández y Baptista (2014), el enfoque cuantitativo “utiliza la recolección de datos para probar hipótesis con base en la medición numérica y el análisis estadístico, con el fin de establecer patrones de comportamiento y probar teorías” (p. 4). En consonancia con esta definición, la presente investigación se orienta a validar de forma objetiva y sistemática una propuesta metodológica para optimizar modelos de IA embebidos, sustentada en evidencia empírica y resultados replicables.

3.4 Diseño

El diseño de investigación, es el plan o la estructura lógica que guía el desarrollo del estudio. Especifica cómo se recolectan, analizan y evalúan los datos, así como la forma en que se manipulan las variables para responder al problema de investigación.

Según Hernández Sampieri et al. (2014), el diseño “constituye el plan o estrategia concebida para obtener la información que se desea” y en investigaciones experimentales implica manipular deliberadamente variables independientes para observar su efecto sobre variables dependientes bajo condiciones controladas.

El diseño de esta investigación es de tipo experimental puro, ya que se cumplen las condiciones esenciales de este tipo de estudios: manipulación intencionada de variables independientes, medición de su efecto en variables dependientes y control riguroso de factores externos que podrían interferir en los resultados.

En este caso, la variable independiente es la técnica de optimización aplicada al modelo de reconocimiento de patrones, y las variables dependientes son el rendimiento del modelo medido en términos de latencia, precisión, velocidad de procesamiento (FPS), uso de memoria, y consumo energético. Se controlarán variables como el tipo de modelo base, el hardware embebido utilizado, las condiciones del entorno y el *dataset* empleado.

Este diseño experimental permite establecer relaciones causales entre la técnica de optimización aplicada y el comportamiento del modelo en condiciones reales de edge computing, lo cual es consistente con el enfoque explicativo y cuantitativo adoptado en esta investigación.

A continuación, se presentan las 10 fases del enfoque cuantitativo, de acuerdo a lo establecido por Hernández Sampieri et al. (2014):

3.4.1 Fase 1: Idea

La idea inicial de esta investigación, surgió a partir del creciente interés en la optimización de modelos de inteligencia artificial que operen de forma eficiente en entornos con recursos computacionales limitados. En particular, se identificó la necesidad de optimizar algoritmos de reconocimiento de patrones para su ejecución en hardware embebido de bajo costo, directamente en el borde (*edge computing*), sin depender de procesamiento externo. Esto se alinea con las tendencias actuales de descentralización computacional, que buscan reducir la latencia y mejorar la eficiencia energética.

La motivación se enmarca en el reto de trasladar modelos complejos de visión por computadora hacia plataformas más accesibles, eficientes y portátiles, como aquellas usadas en robótica educativa o sistemas autónomos. Como caso de aplicación, se plantea la detección visual de robots y delimitaciones en un entorno competitivo tipo sumo bot.

3.4.2 Fase 2: Planteamiento del problema

La creciente demanda por soluciones de IA en dispositivos autónomos, ha generado un interés particular por ejecutar modelos de reconocimiento en hardware embebido de bajo costo. Sin embargo, aún existe una brecha entre los modelos entrenados en entornos de alto rendimiento y su implementación efectiva en plataformas con recursos limitados.

El problema específico que se plantea es: *¿Cómo optimizar los algoritmos de reconocimiento de patrones en hardware embebido de bajo costo, para mejorar el rendimiento y garantizar tiempos de inferencia mínimos en entornos con procesamiento descentralizado y condiciones ambientales variables?*

Esto implica desafíos técnicos relacionados con la eficiencia computacional, la adaptación del modelo al entorno embebido, la minimización del consumo energético y la necesidad de metodologías claras para guiar esta optimización. En este sentido, se propone el desarrollo de un marco metodológico sistematizado que oriente a ingenieros y desarrolladores en este proceso.

3.4.3 Fase 3: Revisión de la literatura y desarrollo del marco teórico

La revisión de literatura se realiza de forma estructurada, utilizando como base la plantilla de Biolchini para guiar la selección y análisis de estudios primarios. Se aplican criterios rigurosos de calidad, como H-Index, SJR y pertenencia a revistas Q1, además de cadenas de búsqueda específicas en bases como IEEE Xplore y ACM Digital Library. Se priorizan estudios recientes (posteriores a 2021) que aborden la

optimización de algoritmos de IA en hardware embebido bajo condiciones de *edge computing*.

Este proceso permite identificar cinco ejes fundamentales que conforman el marco teórico: el reconocimiento de patrones con IA, las técnicas de optimización de modelos, la implementación en sistemas embebidos, la evaluación del desempeño en el borde, y los fundamentos del *edge computing*. Estos ejes sustentan la base conceptual del marco metodológico propuesto.

3.4.4 Fase 4: Visualización del alcance del estudio

El alcance de la investigación es explicativo, ya que busca analizar cómo y por qué las distintas técnicas de optimización afectan el rendimiento de modelos de IA al ser ejecutados en sistemas embebidos. Se pretende establecer relaciones causales entre las estrategias de optimización y las variables de rendimiento (latencia, precisión, uso de memoria, FPS y consumo energético), mediante un diseño experimental riguroso.

3.4.5 Fase 5: Elaboración de hipótesis y definición de variables

Hipótesis principal:

- H1: La técnica de optimización aplicada, influye significativamente en el rendimiento del modelo de inteligencia artificial en términos de latencia, precisión, FPS y consumo de recursos.

Variables:

- Variable independiente: Técnica de optimización aplicada (cuantización, pruning, etc.).
- Variables dependientes: latencia de inferencia, precisión del modelo, FPS, uso de memoria, consumo energético.

- Variables controladas: tipo de modelo, hardware embebido, *dataset*, entorno físico, resolución de entrada.

3.4.6 Fase 6: Desarrollo del diseño de investigación

Se adopta un diseño experimental puro. Se manipula la técnica de optimización del modelo y se mide su efecto en las variables dependientes en condiciones controladas. Todos los experimentos se ejecutan sobre los mismos dispositivos embebidos, el modelo base y el *dataset*, asegurando condiciones homogéneas. Esto permite comparar los efectos de cada técnica de manera objetiva y replicable.

3.4.7 Fase 7: Definición y selección de la muestra

La muestra fue seleccionada intencionalmente y está compuesta por:

- Dispositivos embebidos: Nicla Vision, Portenta H7 Vision Bundle.
- *Dataset* visual del dojo - robots, capturado en condiciones estandarizadas para evaluar el rendimiento del modelo en escenarios representativos.

3.4.8 Fase 8: Recolección de los datos

Se implementan modelos optimizados en los dispositivos embebidos y se registran las siguientes métricas:

- Latencia (ms)
- Precisión (%)
- FPS
- Uso de memoria y CPU
- Consumo energético (si aplica) Los datos se obtienen mediante *scripts*, herramientas de medición embebida y observación directa.

3.4.9 Fase 9: Análisis de los datos

Se organiza la información en tablas y se analizan los resultados mediante estadística descriptiva. Se evalúa el impacto de cada técnica de optimización sobre el

rendimiento del modelo, comparando los resultados entre configuraciones y validando la hipótesis planteada.

3.4.10 Fase 10: Elaboración del reporte de resultados

Se elabora el informe final que incluye el marco metodológico desarrollado, los resultados experimentales, las recomendaciones técnicas, y una reflexión sobre las limitaciones y posibilidades futuras del enfoque propuesto para la optimización de IA en el borde.

3.5 Población y Muestreo

La población de esta investigación, está conformada por modelos de inteligencia artificial de reconocimiento de patrones susceptibles de ser optimizados para su ejecución en hardware embebido de bajo costo. Esta población incluye modelos livianos (por ejemplo, YOLOv5n, MobileNet, EfficientDet-lite) que puedan ser adaptados mediante técnicas de optimización como la cuantización, el *pruning* o la reducción de resolución.

El muestreo es de tipo intencional y se realiza sobre modelos de IA seleccionados específicamente para cumplir con los siguientes criterios:

- Ser compatibles con *frameworks* de optimización (TFLite, ONNX, etc.).
- Poder ejecutarse en tiempo real en hardware embebido con recursos limitados.
- Tener documentación, comunidad activa y capacidad de medición de métricas clave.

Adicionalmente, se utilizaron dispositivos embebidos específicos como entorno de validación experimental, y un *dataset* controlado (dojo – robots) como insumo constante para pruebas.

Capítulo 4. Análisis de la situación

En el presente capítulo, se expone el análisis de la situación actual en torno al uso de sistemas embebidos de bajo costo en el borde (*edge computing*), para el reconocimiento de patrones visuales mediante algoritmos de inteligencia artificial aplicados a tareas de visión por computadora.

El análisis se fundamenta en dos fuentes principales: una revisión de literatura científica reciente y una observación técnica basada en el estudio de documentación especializada y características funcionales de los dispositivos seleccionados.

El propósito del análisis es identificar las capacidades reales, limitaciones operativas y condiciones tecnológicas que deben considerarse para el desarrollo de un marco metodológico orientado a la implementación eficiente de inteligencia artificial embebida en contextos con recursos limitados. Los hallazgos permiten sustentar técnicamente la viabilidad de dicha propuesta y orientar su diseño hacia aplicaciones prácticas como las competencias de sumo bot.

4.1 Instrumentos de recolección de información

Dado que esta investigación aborda una problemática emergente desde una perspectiva técnica y contextual, se emplearon dos instrumentos principales para la recolección de información: la revisión documental y la observación técnica. La revisión documental permite identificar avances y tendencias relevantes en el uso de inteligencia artificial embebida en sistemas de bajo costo. Por su parte, la observación técnica, de carácter analítico y no experimental, se centró en el estudio sistemático de documentación especializada y fuentes técnicas sobre los dispositivos seleccionados. Ambos instrumentos fueron elegidos por su idoneidad metodológica y por su capacidad para aportar insumos fundamentales desde el enfoque científico, tecnológico y aplicado que guía esta investigación.

4.1.1 Revisión documental

La revisión de artículos científicos recientes permitió identificar avances clave en el campo de la inteligencia artificial embebida, particularmente en microcontroladores para aplicaciones de visión por computadora y *edge computing*. Este análisis documental se enfoca en soluciones que permiten ejecutar inferencias de redes neuronales profundas en dispositivos con capacidades extremadamente limitadas, así como en la creciente relevancia de estos sistemas en el contexto tecnológico actual.

4.1.1.1 Aumento de la importancia de los sistemas embebidos trabajando en el borde

La adopción de inteligencia artificial embebida en dispositivos de borde, ha experimentado un crecimiento notable en los últimos años. Zhang y Li (2023) destacan que la inteligencia embebida está desempeñando un papel fundamental en la transformación digital, particularmente en sectores como la manufactura inteligente, automatización industrial y dispositivos móviles. El desarrollo de arquitecturas de hardware eficientes y modelos optimizados ha permitido implementar capacidades de aprendizaje automático directamente en dispositivos de bajo consumo, ampliando las posibilidades de uso en el mundo real.

Asimismo, Shuvo et al. (2023) enfatizan que la migración del procesamiento en la nube hacia el borde se está acelerando debido a beneficios como la baja latencia, la eficiencia energética y la mejora en la privacidad. Según estos autores, muchas aplicaciones que tradicionalmente requerían conexión continua a servidores ahora pueden operar de forma autónoma y confiable desde microcontroladores integrados, lo cual ha abierto nuevas oportunidades en campos como salud, vigilancia, agricultura de precisión y manufactura avanzada

4.1.1.2 TensorFlow Lite Micro (TFLM): portabilidad y eficiencia en TinyML

El *framework* TensorFlow Lite Micro (TFLM) se posiciona como una solución altamente flexible para llevar modelos de *machine learning* a microcontroladores. Diseñado con un enfoque interpretado, TFLM facilita la portabilidad entre diferentes arquitecturas embebidas sin necesidad de generar código específico para cada plataforma. Según Duke et al. (2021), TFLM se caracteriza por su estructura modular, su compatibilidad con compiladores cruzados y su capacidad de ejecución en dispositivos con menos de 16 KB de RAM, haciéndolo ideal para aplicaciones TinyML en entornos reales.

4.1.1.3 MCUNet: modelos compactos con alto rendimiento en dispositivos limitados

El trabajo de Lin et al. (2020) propone MCUNet, una combinación innovadora de TinyNAS y TinyEngine que permite lograr inferencias de alto rendimiento en microcontroladores comerciales. MCUNet alcanzó más del 70% de precisión en ImageNet utilizando hasta cinco veces menos memoria Flash y SRAM que MobileNetV2, marcando un hito en la ejecución de modelos complejos en hardware de bajo costo

4.1.1.4 MicroNets: modelos optimizados por búsqueda de arquitectura diferenciable

Banbury et al. (2021) introducen MicroNets, una familia de modelos desarrollados mediante búsqueda de arquitectura neuronal diferenciable (DNAS). Estas redes están diseñadas específicamente para cumplir con restricciones de latencia y consumo energético en microcontroladores. MicroNets alcanza resultados de última generación en tareas como detección de palabras clave, reconocimiento de patrones visuales y detección de anomalías, todos ejecutables con TensorFlow Lite Micro.

4.1.1.5 TinyTL: eficiencia de memoria sin sacrificar precisión

Finalmente, el enfoque TinyTL propone una estrategia para mantener los parámetros del modelo congelados durante el entrenamiento, modificando solo capas adaptativas ligeras. Según Cai et al. (2020), esto permite realizar aprendizaje en el dispositivo (*on-device learning*) utilizando una fracción de la memoria, sin afectar significativamente la precisión del modelo, lo cual representa un avance clave hacia sistemas autoaprendientes en el borde.

4.1.1.6 Conclusión de la revisión documental

La revisión de literatura científica reciente, confirma el crecimiento sostenido y estratégico del uso de inteligencia artificial embebida en el borde (*edge computing*), impulsado por la necesidad de procesamiento autónomo, eficiente y en tiempo real. En algunos de los artículos analizados, destacan que los sistemas embebidos están dejando de ser elementos periféricos para convertirse en plataformas clave en la ejecución de modelos de aprendizaje automático, especialmente en escenarios con recursos limitados.

Se identificaron soluciones tecnológicas robustas como TensorFlow Lite Micro, que facilita la portabilidad de modelos en microcontroladores; arquitecturas como MCUNet y MicroNets, que optimizan el uso de memoria y procesamiento sin comprometer precisión; y enfoques como TinyTL, que habilitan el aprendizaje adaptativo local sin requerimientos de entrenamiento intensivo. Estas propuestas no solo evidencian el potencial técnico actual, sino que también abren la puerta al diseño de marcos metodológicos replicables, capaces de llevar la inteligencia embebida a contextos educativos, industriales y de innovación aplicada.

En conjunto, los hallazgos respaldan firmemente la viabilidad de implementar soluciones de visión por computadora e IA en dispositivos embebidos de bajo costo, y

justifican el desarrollo de un marco metodológico como el que plantea esta investigación.

4.1.2 Observación Técnica

La observación técnica, constituye un método de recolección de información que permite examinar de forma sistemática las características, funcionamiento y capacidades de dispositivos tecnológicos, a partir del análisis de documentación técnica especializada. En esta investigación, se emplea como herramienta para comprender el potencial y las limitaciones de sistemas embebidos de bajo consumo energético en el contexto de la inteligencia artificial aplicada a visión por computadora.

A diferencia de una observación basada en pruebas experimentales, esta observación se apoya en el estudio de fichas técnicas, guías de integración, literatura científica y reportes de desempeño de los dispositivos seleccionados, lo cual permite extraer datos objetivos sobre variables técnicas clave como capacidad de procesamiento, compatibilidad con *frameworks* de aprendizaje automático, uso de memoria y nivel de integración práctica.

4.1.2.1 Objetivo de la observación técnica

El objetivo de esta observación técnica es analizar desde una perspectiva documental y analítica, las capacidades declaradas y potenciales de ejecución de inteligencia artificial embebida en dos dispositivos representativos de bajo costo: Nicla Vision y Portenta H7 Vision Bundle. Esta evaluación se enfoca en aspectos técnicos relevantes para tareas de visión por computadora en el borde, tales como: procesamiento de imágenes, compatibilidad con *frameworks* optimizados (como TensorFlow Lite Micro, OpenMV y Edge Impulse), consumo de memoria y facilidad de integración en aplicaciones educativas o competitivas.

Asimismo, esta observación busca contrastar las capacidades reportadas por los fabricantes con las necesidades técnicas del caso de estudio (sumo bot), a fin de identificar fortalezas, limitaciones y requerimientos mínimos para el diseño de un marco metodológico aplicable y replicable en contextos reales.

4.1.2.2 Dispositivos observados

Como parte de esta observación técnica se seleccionaron dos dispositivos embebidos de bajo consumo energético y alta eficiencia: Nicla Vision y Portenta H7 con Vision Shield. Ambos forman parte del ecosistema Arduino Pro y han sido diseñados específicamente para aplicaciones de inteligencia artificial en el borde (*edge AI*), incluyendo visión por computadora, aprendizaje automático embebido y procesamiento distribuido.

La selección de estos dispositivos responde a su potencial para ejecutar modelos de inferencia localmente, sin depender de infraestructura externa, lo que los hace especialmente adecuados para entornos educativos, prototipado rápido y competencias como el sumo bot. A continuación, se presenta una tabla comparativa basada en las especificaciones oficiales de sus respectivos *datasheets*:

Característica	**Nicla Vision**	**Portenta H7 con Vision Shield**
Microcontrolador	STM32H747AII6	STM32H747XI
Velocidad de núcleos M7/M4	480 MHz / 240 MHz	480 MHz / 240 MHz
RAM interna	1 MB	1 MB
Flash interna	2 MB	2 MB
Memoria externa (SDRAM / QSPI)	16 MB Flash externa	8 MB SDRAM / 16 MB QSPI Flash

Cámara integrada	Sí	Sí (con Vision Shield)
Resolución de cámara	2 MP (GC2145)	HM-01B0 (320x320)
Conectividad inalámbrica	Wi-Fi 802.11 b/g/n, Bluetooth	Wi-Fi, BLE, Ethernet (según Shield)
Frameworks compatibles	OpenMV, Edge Impulse	TensorFlow Lite Micro, Edge Impulse
Consumo en captura de imagen	105 mA	230 mA (estimado con ambos núcleos activos)
Consumo en deep sleep (típico)	374 µA	0.67 mA
Interfaz de programación	OpenMV IDE, Arduino IDE	Arduino IDE, Mbed OS, CMSIS
Tamaño	22.86 mm x 22.86 mm	102 mm x 25 mm
Ventaja destacada	Compacta, con cámara integrada y bajo consumo	Mayor capacidad de cómputo y expansión
Limitación destacada	Limitada en procesamiento y almacenamiento	Curva de configuración más alta y dependiente del Shield

Tabla 13. Tabla comparativa Nicla Vision Vs Portenta H7. Fuente: elaboración propia.

Esta comparación técnica permite visualizar las fortalezas y limitaciones relativas de cada plataforma. La plataforma Nicla Vision, con su diseño compacto y cámara integrada, destaca por su bajo consumo energético y facilidad de integración en proyectos rápidos. En contraste, la plataforma Portenta H7, junto con el Vision Shield, ofrece una arquitectura más potente y escalable, aunque requiere mayor configuración y conocimientos técnicos para su implementación.

Esta diferencia sugiere que la plataforma Nicla Vision es ideal para tareas de clasificación o reconocimiento visual simple, mientras que la plataforma Portenta H7 es más adecuada para aplicaciones complejas o personalizables que requieran múltiples capas de procesamiento o detección simultánea de objetos.

4.1.3 Justificación del marco metodológico

El análisis de la situación realizado mediante revisión documental y observación técnica, evidencia una oportunidad concreta para el desarrollo de un marco metodológico que facilite la implementación de sistemas de visión por computadora con inteligencia artificial embebida, especialmente en plataformas de bajo consumo y recursos limitados como Nicla Vision y Portenta H7.

La literatura especializada coincide en destacar el crecimiento de soluciones de aprendizaje automático ejecutadas en el borde (TinyML), como respuesta a la necesidad de sistemas más autónomos, eficientes y descentralizados. Sin embargo, también se identifican desafíos recurrentes: desde restricciones técnicas en hardware hasta la falta de rutas claras para integrar modelos de IA optimizados en microcontroladores. Estas condiciones generan una barrera significativa para su adopción en contextos educativos, competitivos o de innovación rápida.

Por otra parte, la observación técnica permitió delimitar con precisión las capacidades reales, limitaciones y condiciones de uso de los dispositivos analizados. Si bien estos ofrecen el potencial para ejecutar modelos de inferencia local, su correcto aprovechamiento requiere de un proceso metodológico que oriente decisiones clave como:

- Selección del dispositivo según la tarea.
- Definición del tipo de modelo a emplear.
- Herramientas de desarrollo y frameworks compatibles.

- Estrategias de optimización necesarias para cada caso.
- Validación en entornos reales como el sumo bot.

Ante este panorama, el desarrollo de un marco metodológico estructurado, modular y replicable permitirá reducir la complejidad de implementación, estandarizar el flujo de trabajo, y ampliar la aplicabilidad de este tipo de soluciones tanto en el ámbito educativo como competitivo. Dicho marco no solo facilitará la adopción de tecnologías emergentes en sistemas embebidos, sino que también aportará valor pedagógico al promover el aprendizaje activo mediante la experimentación con hardware real y modelos inteligentes optimizados para el borde.

Capítulo 5. Propuesta de Solución

Este capítulo será completado en conjunto con el tutor académico en el siguiente curso PIA-02.

Capítulo 6. Conclusiones y Recomendaciones

Este capítulo será completado en conjunto con el tutor académico en el siguiente curso PIA-02.

Bibliografía

Baller, S. P., Jindal, A., Chadha, M., & Gerndt, M. (2021). DeepEdgeBench: Benchmarking Deep Neural Networks on Edge Devices. arXiv preprint arXiv:2108.09457. https://arxiv.org/abs/2108.09457

Banbury, C. R., Zhou, C., Fedorov, I., Matas Navarro, R., Thakker, U., Gope, D., Janapa Reddi, V., Mattina, M., & Whatmough, P. N. (2021). MicroNets: Neural network architectures for deploying TinyML applications on commodity microcontrollers. Proceedings of the 4th Conference on Machine Learning and Systems (MLSys). https://arxiv.org/abs/2010.11267

Blanco-Filgueira, B., García-Lesta, D., Fernández-Sanjurjo, M., Brea, V. M., & López, P. (2021). Deep Learning-Based Multiple Object Visual Tracking on Embedded System for IoT and Mobile Edge Computing Applications. IEEE Internet of Things Journal, 8(4), 2541–2552. https://doi.org/10.1109/JIOT.2020.3015841

Cai, H., Gan, C., Zhu, L., & Han, S. (2020). TinyTL: Reduce memory, not parameters for efficient on-device learning. Advances in Neural Information Processing Systems, 33, 11285–11297. https://arxiv.org/abs/2007.11622

David, R., Duke, J., Jain, A., Reddi, V. J., Jeffries, N., Li, J., Kreeger, N., Nappier, I., Natraj, M., Regev, S., Rhodes, R., Wang, T., & Warden, P. (2021). TensorFlow Lite Micro: Embedded machine learning on TinyML systems. Proceedings of the 4th Conference on Machine Learning and Systems (MLSys). https://arxiv.org/abs/2010.08678

Feronato, J., Beltrán, V., & Tolosana-Calasanz, R. (2023). An Energy-Aware Approach to Design Self-Adaptive AI-based Applications on the Edge. arXiv preprint arXiv:2309.00022. https://arxiv.org/abs/2309.00022

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

Han, D., Liu, Y., Xie, H., Zhang, Y., & Zhang, J. (2020). KubeEdge.AI: AI Platform for Edge Devices. arXiv preprint arXiv:2007.09227. https://arxiv.org/abs/2007.09227

Hao, C., Dotzel, J., Xiong, J., Benini, L., Zhang, Z., & Chen, D. (2021). Enabling Design Methodologies and Future Trends for Edge AI: Specialization and Co-design. arXiv preprint arXiv:2103.15750.

Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, P. (2014). Metodología de la investigación (6.ª ed.). McGraw-Hill.

Hoefler, T., Alistarh, D., Ben-Nun, T., Dryden, N., & Peste, A. (2021). Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks. arXiv preprint arXiv:2102.00554.

keywords: {Quantization (signal);Signal to noise ratio;Receivers;Bandwidth;White noise;Timing;Gaussian noise;Edge computing;cloud-computing;datacenter;SDN;NFV;C-RAN;IoT},

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

Li, Y., & Shami, A. (2020). On Hyperparameter Optimization of Machine Learning Algorithms: Theory and Practice. arXiv preprint arXiv:2007.15745.

Lin, J., Chen, W.-M., Lin, Y., Cohn, J., Gan, C., & Han, S. (2020). MCUNet: Tiny deep learning on IoT devices. Advances in Neural Information Processing Systems, 33, 11711–11722. https://arxiv.org/abs/2007.10319

Liu, D., Kong, H., Luo, X., Liu, W., & Subramaniam, R. (2020). Bringing AI To Edge: From Deep Learning's Perspective. arXiv preprint arXiv:2011.14808. https://arxiv.org/abs/2011.14808

Liu, Y., & Zhang, Y. (2023). A Review of Artificial Intelligence in Embedded Systems. Journal of Embedded Systems, 15(2), 123-145.

Nagel, M., Fournarakis, M., Amjad, R. A., Bondarenko, Y., van Baalen, M., & Blankevoort, T. (2021). A White Paper on Neural Network Quantization. arXiv preprint arXiv:2106.08295.

Noergaard, T. (2005). Embedded systems architecture: A comprehensive guide for engineers and programmers. Newnes

Novac, P.-E., Boukli Hacene, G., Pegatoquet, A., Miramond, B., & Gripon, V. (2021). Quantization and Deployment of Deep Neural Networks on Microcontrollers. IEEE Access, 9, 136962–136973. https://doi.org/10.1109/ACCESS.2021.3117326

Russell, S., & Norvig, P. (2010). Artificial Intelligence: A Modern Approach (3rd ed.). Prentice Hall.

Sharma, A., & Kaur, R. (2013). A Survey on Artificial Intelligence Algorithms. International Journal of Computer Applications, 76(9), 7-10.

Shuvo, M. M. I., Sarkar, M. S. K., Raihan, M. A., Ferdous, M. Z., Khondoker, R., Momen, S., & Kaiser, M. S. (2023). Efficient acceleration of deep learning inference on resource-constrained edge devices: A review. IEEE Access, 11, 48525–48542.

Wan, Z., Lele, A. S., & Raychowdhury, A. (2022). Circuit and System Technologies for Energy-Efficient Edge Robotics. IEEE Transactions on Circuits and Systems I: Regular Papers, 69(4), 1607–1620. https://doi.org/10.1109/TCSI.2022.3140423

Widrow, B. (1964). Pattern recognition and adaptive control. IEEE Transactions on Applications and Industry, 83, 269–285. https://isl.stanford.edu/~widrow/papers/j1964patternrecognition.pdf

Y. Zhao, W. Wang, Y. Li, C. Colman Meixner, M. Tornatore and J. Zhang, "Edge Computing and Networking: A Survey on Infrastructures and Applications," in IEEE Access, vol. 7, pp. 101213-101230, 2019, doi: 10.1109/ACCESS.2019.2927538.

Zhang, Z., & Li, J. (2023). A review of artificial intelligence in embedded systems. Micromachines, 14(4), 897.