COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO DE FIN DE GRADO

Optimización de Redes de Sensores y Actuadores para Aplicaciones IoT con Inteligencia Artificial.

Autor: Aurora Junco Miralles

Director: Raúl Robledo Cabezuela

Madrid

OPTIMIZACIÓN DE REDES DE SENSORES Y ACTUADORES PARA APLICACIONES IOT CON INTELIGENCIA ARTIFICIAL

Autor: Junco Miralles, Aurora.
Director: Robledo Cabezuela, Raúl.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

Este proyecto busca desarrollar un sistema de optimización de redes de sensores y actuadores en entornos IoT industriales mediante técnicas de Inteligencia Artificial. El objetivo principal es mejorar la eficiencia de la transmisión de datos y optimizar la toma de decisiones en tiempo real, centrándose en los retos que presentan infraestructuras como plantas fotovoltaicas. El sistema combina sensores ADXL345 conectados a microcontroladores ESP32, comunicación mediante LoRaWAN y MQTT, y modelos de Machine Learning para la detección de eventos y la predicción de respuestas de control.

Los resultados obtenidos demuestran las posibles mejoras significativas en la eficiencia energética, precisión en la detección de eventos anómalos y una notable reducción del tráfico de datos transmitido, contribuyendo a redes de sensores más sostenibles e inteligentes.

Palabras clave: IoT, Inteligencia Artificial, MQTT, LoRaWAN, Machine Learning, Optimización de redes

1. Introducción

El Internet de las Cosas (IoT) forma actualmente un papel notable en la transformación digital de sectores como la energía y la industria. Sin embargo, la gestión eficiente de datos generados por sensores y actuadores plantea importantes desafíos, especialmente en términos de eficiencia energética y latencia en la toma de decisiones.

Este proyecto aborda estos retos proponiendo un modelo que integra Inteligencia Artificial con protocolos de comunicación de bajo consumo como LoRaWAN y MQTT, permitiendo una transmisión selectiva de datos basada en la detección de eventos relevantes y la optimización de la respuesta de los actuadores. El enfoque desarrollado se aplica a un caso representativo: la monitorización y ajuste de estructuras de plantas fotovoltaicas, donde las condiciones ambientales requieren respuestas rápidas y precisas para mantener la eficiencia operativa.

2. Definición del proyecto

El objetivo del proyecto es diseñar un sistema que minimice las transmisiones innecesarias y prediga de forma precisa la señal de control a aplicar en respuesta a eventos detectados. Para ello, se estructura el trabajo en dos partes principales:

- Detección inteligente de eventos: Se utiliza un modelo de aprendizaje no supervisado (Isolation Forest) combinado con el método de Otsu para etiquetar automáticamente eventos anómalos sin necesidad de un umbral manual.

- Predicción de la respuesta PWM (Pulse Width Modulation): Se entrena un modelo de regresión supervisado (HistGradientBoosting) sobre las muestras clasificadas como eventos para predecir el valor de PWM que debe enviarse al actuador.

Ambas estrategias se integran en una arquitectura modular, diseñada para ser escalable y adaptable a diversas infraestructuras IoT.

3. Descripción del sistema

El sistema desarrollado se organiza en tres capas, partiendo de una serie de datos simulados a partir de una situación real y permitiendo crear las tres capas del sistema.

- Capa de sensores y actuadores: Basada en sensores de aceleración ADXL345 acoplados a microcontroladores ESP32, encargados de captar inclinaciones y transmitir solo datos relevantes.
- Capa de comunicación: Implementada mediante los protocolos LoRaWAN y MQTT, que permiten una transmisión eficiente y de bajo consumo energético.
- Capa de Inteligencia Artificial: Encargada de la detección de eventos mediante Isolation Forest y de la predicción de la respuesta que calcula el valor de PWM óptimo a través del algoritmo HistGradientBoosting, mejorando así la eficiencia del sistema.

Este diseño se ha simulado en un entorno de Simulink, con la capa de Inteligencia Artificial desarrollada en Python, permitiendo validar el flujo completo desde la captura de datos hasta la respuesta final del actuador.

4. Resultados

Se ha logrado desarrollar un sistema que permite una reducción del tráfico de datos transmitido, al activar la comunicación únicamente ante eventos detectados o mediante mensajes keepalive periódicos.

La predicción de la señal PWM ha mostrado un error absoluto medio (MAE) de 7.57 y un coeficiente de determinación (R^2) de 0.611 en el subconjunto utilizado como test.

La simulación integrada en Simulink ha confirmado un MAE aún mejor de 3.95 y un R^2 de 0.742 sobre una simulación realizada en los primeros 10.000 puntos evaluados.

Estos resultados validan la eficiencia del sistema tanto en la detección temprana de eventos como en la generación de respuestas adaptativas, optimizando el consumo energético y la fiabilidad del sistema. En la Figura 22 a continuación se muestra gráficamente la comparativa entre el PWM real y las predicciones, consiguiendo destacar la capacidad de ajuste del modelo a lo largo de todo el rango de valores de PWM.

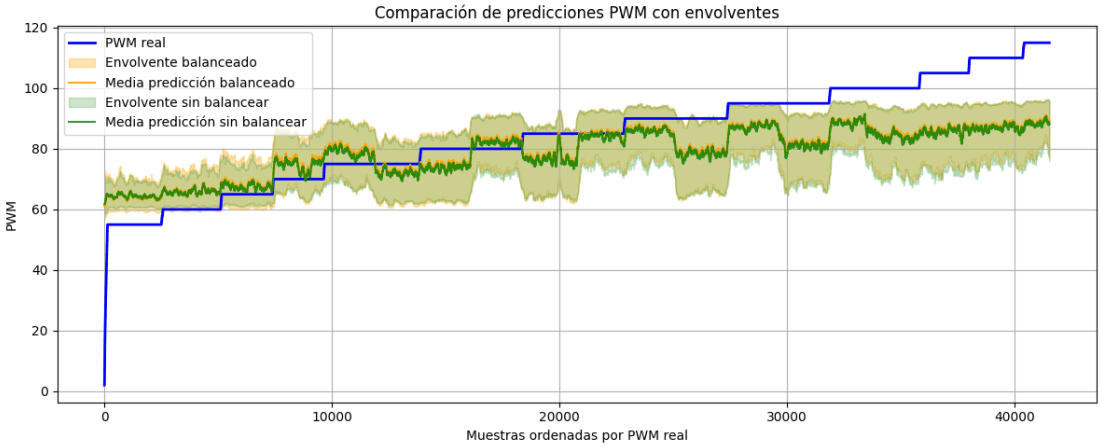


Figura 22: Predicción de PWM ordenado por orden creciente

5. Conclusiones

El proyecto demuestra que los algoritmos de Inteligencia Artificial son capaces de optimizar redes de sensores IoT en aplicaciones industriales como las plantas fotovoltaicas. Se consigue una red más eficiente, capaz de anticipar eventos críticos y actuar con rapidez, contribuyendo en muchos aspectos como la sostenibilidad.

Además, la arquitectura propuesta es modular y adaptable, lo que permite plantear su futura aplicación en otros sectores industriales o urbanos. Como áreas de mejora o trabajos futuros, se proponen ideas como ser capaces de aplicar la IA desde el microcontrolador, para ser capaces de ajustar dinámicamente la detección de eventos ante cambios en las condiciones, así como implementar estrategias de solución de fallos en la comunicación como la caída de algún sensor, mejorando aún más la capacidad de adaptación de la red.

OPTIMIZATION OF SENSOR AND ACTUATOR NETWORKS FOR IOT APPLICATIONS WITH ARTIFICIAL INTELLIGENCE

Author: Junco Miralles, Aurora.
Supervisor: Robledo Cabezuela, Raúl.
Collaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

This project aims to develop a system for optimizing sensor and actuator networks in industrial IoT environments using Artificial Intelligence techniques. The main objective is to improve data transmission efficiency and optimize real-time decision-making, focusing on the challenges presented by infrastructures such as photovoltaic plants. The system combines ADXL345 sensors connected to ESP32 microcontrollers, communication via LoRaWAN and MQTT, and Machine Learning models for event detection and control response prediction.

The results demonstrate significant potential improvements in energy efficiency, accuracy in detecting anomalous events, and a notable reduction in transmitted data traffic, contributing to more sustainable and intelligent sensor networks.

Keywords: IoT, Artificial Intelligence, MQTT, LoRaWAN, Machine Learning, Network Optimization

1. Introduction

The Internet of Things (IoT) currently plays a major role in the digital transformation of sectors such as energy and industry. However, efficiently managing the large volumes of data generated by sensors and actuators poses significant challenges, particularly regarding energy efficiency and decision-making latency.

This project addresses these challenges by proposing a model that integrates Artificial Intelligence with communication protocols such as LoRaWAN and MQTT, enabling selective data transmission based on the detection of relevant events and optimization of actuator responses. The approach is applied to a representative case which is monitoring and adjusting structures in photovoltaic plants, where environmental conditions require quick and precise responses to maintain operational efficiency.

2. Project Definition

The goal of the project is to design a system that minimizes unnecessary transmissions and accurately predicts the control signal to be applied in response to detected events. The work is structured into two main components:

- Intelligent event detection: An unsupervised learning model (Isolation Forest) is combined with the Otsu method to automatically label anomalous events without the need for manual threshold setting.

- PWM (Pulse Width Modulation) response prediction: A supervised regression model (HistGradientBoosting) is trained on the samples classified as events to predict the PWM value to be sent to the actuator.

Both strategies are integrated into a modular architecture, designed to be scalable and adaptable to various IoT infrastructures.

3. System Description

The developed system is organized into three layers, based on a dataset simulated from real-world conditions, enabling the creation of the system's layers:

- Sensor and actuator layer: Based on ADXL345 accelerometers connected to ESP32 microcontrollers, responsible for detecting inclinations and transmitting only relevant data.
- Communication layer: Implemented using LoRaWAN and MQTT protocols, enabling efficient and low power data transmission.
- Artificial Intelligence layer: Responsible for event detection through Isolation Forest and optimal PWM response prediction via HistGradientBoosting, improving the system´s adaptability.

This design has been simulated in a Simulink environment, with the AI layer developed in Python, allowing validation of the complete flow from the origin of the data to the actuator's final response.

4. Results

The system developed successfully reduces transmitted data traffic by activating communication only upon event detection or through periodic keepalive messages.

The PWM signal prediction achieved a Mean Absolute Error (MAE) of 7.57 and a coefficient of determination (R^2) of 0.611 on the test set.

The integrated Simulink simulation confirmed an even better MAE of 3.95 and an R^2 of 0.742 over the first 10,000 evaluated points.

These results validate the system's efficiency both in early event detection and in generating adaptive responses, optimizing energy consumption and system reliability. Figure 22 below graphically shows the comparison between the real PWM and the model predictions, highlighting the model's adjustment capability across the PWM value range.

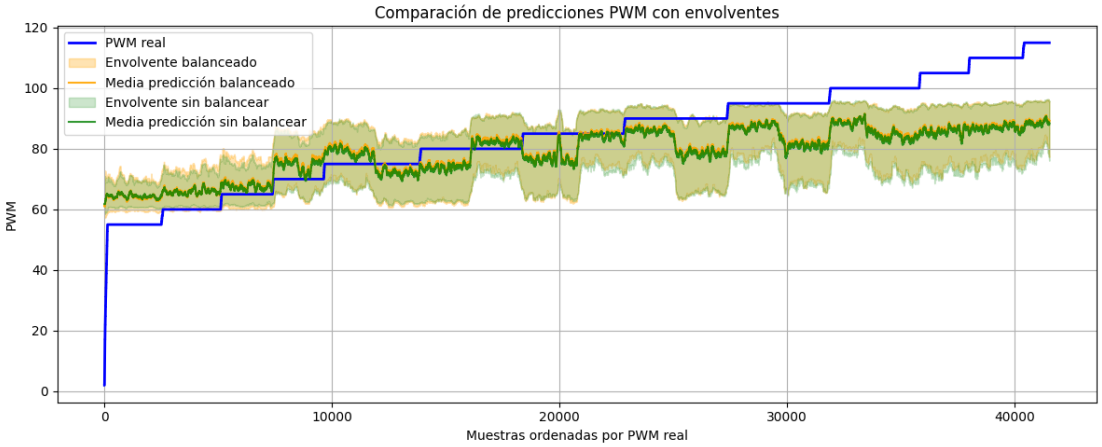


Figure 22: PWM prediction sorted by increasing order

5. Conclusions

The project demonstrates that Artificial Intelligence algorithms can effectively optimize IoT sensor networks in industrial applications such as photovoltaic plants. A more efficient network is achieved, capable of anticipating critical events and reacting quickly, contributing to sustainability.

In addition, the proposed architecture is modular and adaptable, allowing its future application in other industrial or urban sectors. Potential improvements and future work include deploying AI capabilities directly on microcontrollers to dynamically adjust event detection to changing conditions and implementing strategies in case of default, such as handling sensor communication failures, further enhancing the network's capacity of adapting.

Índice de la memoria

Índice de figuras

Índice de tablas

1. INTRODUCCIÓN

Tal como se ha introducido en el resumen previo, donde se ha presentado de manera muy general el trabajo a desarrollar, en este capítulo se detallarán con mayor precisión los aspectos fundamentales del proyecto. Se expondrá la motivación que ha llevado a su planteamiento, los objetivos que se persiguen, la metodología a emplear y el enfoque de desarrollo previsto para llevarlo a cabo de manera efectiva.

MOTIVACIÓN DEL PROYECTO

Este proyecto surge a partir del interés por optimizar y modernizar los procesos industriales mediante la integración de distintas disciplinas dentro del ámbito de las telecomunicaciones e Internet de las Cosas (IoT). Una de las áreas clave de la Ingeniería de Telecomunicaciones, la cual está muy presente en este proyecto, es la arquitectura de redes, un campo en constante evolución y con relevancia actual en múltiples sectores.

Dentro de esta evolución, el desarrollo y proliferación de dispositivos IoT han permitido ampliar significativamente las posibilidades de automatización y monitoreo en tiempo real. Estos dispositivos, cuando se combinan con redes de sensores y actuadores, pueden desempeñar un papel crucial en la optimización de procesos en entornos industriales y comerciales.

El auge de la Inteligencia Artificial y los algoritmos de aprendizaje automático (Machine Learning) abre nuevas oportunidades para la automatización avanzada, permitiendo que las decisiones sean tomadas de manera autónoma y eficiente con base en los datos recopilados por estas redes de sensores. Este enfoque no solo representa una ventaja en términos de optimización industrial, sino que también puede tener un impacto significativo en el ámbito económico y social.

PLANTEAMIENTO

El enfoque de este proyecto se basa inicialmente en un estudio del estado del arte de los protocolos de comunicación más utilizados en entornos IoT, con especial atención a aquellos diseñados para operar en grandes escalas. La selección de estos protocolos estará basada en su capacidad para gestionar comunicaciones eficientes entre sensores distribuidos en áreas amplias, asegurando una transmisión de datos fiable y con el menor consumo energético posible.

El propósito de este estudio previo es identificar los protocolos más adecuados para garantizar la conectividad entre los sensores IoT y la infraestructura de red, permitiendo el envío de datos a una plataforma en la web donde serán analizados. A partir de este análisis,

se generarán respuestas automatizadas que se enviarán a una serie de actuadores, cerrando el ciclo de comunicación y control.

Para ello, es fundamental examinar y comparar distintos protocolos de transmisión en términos de eficiencia, escalabilidad y compatibilidad con los estándares actuales de la industria. Se evaluará también la manera más vaible y eficiente de integrar inteligencia artificial en el procesamiento de los datos transmitidos, con el objetivo de mejorar la toma de decisiones y optimizar el rendimiento de la red.

OBJETIVOS DEL PROYECTO

Tal como se ha explicado, el principal objetivo de este proyecto es diseñar una arquitectura de red de sensores y actuadores basada en IoT que permita la recopilación y transmisión de datos en tiempo real. Esta red será optimizada mediante el uso de algoritmos de Inteligencia Artificial creados a partir de Machine Learning, con el fin de mejorar su capacidad de respuesta ante diversas situaciones detectadas por los sensores.

Entre los objetivos específicos del proyecto se incluyen:

- Evaluar los principales protocolos de comunicación IoT existentes y determinar cuáles son los más adecuados para redes de sensores a gran escala.
- Diseñar una estructura de red eficiente que permita la comunicación fluida entre sensores, actuadores y la plataforma de procesamiento de datos en la web.
- Implementar algoritmos para optimizar el procesamiento de los datos capturados y predecir posibles eventos en base a patrones detectados en la información recopilada.
- Plantear una solución adaptable a entornos industriales, donde la implementación de redes de sensores y actuadores pueda generar mejoras significativas en la automatización de procesos.
- Diseñar una simulación que permita evaluar el comportamiento de la red propuesta y validar su desempeño en distintos escenarios.

Dado que el enfoque del proyecto está en la optimización del sistema de comunicaciones y el procesamiento de datos dentro de la red, se planteará una estructura simplificada que permita analizar el impacto de los algoritmos en la eficiencia del sistema, por lo que este proyecto estará menos centrado en las especificaciones del diseño de la red de sensores y actuadores y más en su actual funcionamiento.

En la Figura 1 se presenta un esquema muy generalizado de la arquitectura de la red propuesta, en la que se analizarán cada uno de los elementos que permite la transmisión de información:

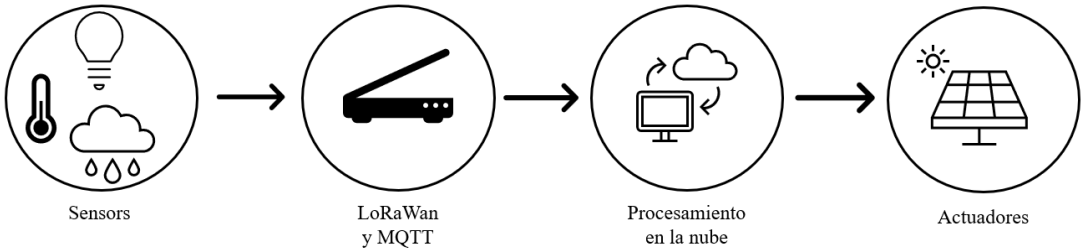


Figura 1: Diagrama simplificado de Arquitectura del Sistema

Como se observa en la Figura 1, el correcto funcionamiento de la red requiere la implementación de dos protocolos distintos: uno responsable de la comunicación con la red IoT para la recopilación y transmisión de datos, y otro encargado de la retransmisión hacia la web, donde los datos serán procesados por el algoritmo de inteligencia artificial. Este algoritmo se encargará de tomar decisiones o emitir alertas para garantizar el mejor desempeño posible de la red.

DESARROLLO

El desarrollo del proyecto se basará en estructura que incluye varias etapas clave. Inicialmente, se llevará a cabo un análisis exhaustivo de las soluciones existentes en el campo de las comunicaciones IoT y la optimización de redes de sensores. Este estudio se encuentra detallado en el Capítulo 2 más adelante y servirá para establecer un marco de referencia para el diseño del sistema.

A partir de esta revisión del estado del arte, se procederá con la selección de protocolos de comunicación y el diseño de la arquitectura de red. Posteriormente, se trabajará en la implementación y simulación del sistema, con el objetivo de evaluar su rendimiento y optimizar su funcionamiento mediante técnicas de Machine Learning.

Uno de los aspectos fundamentales en el desarrollo será garantizar que la solución propuesta aporte un valor añadido respecto a los estudios previos. Para ello, se buscará no solo mejorar la eficiencia de la red de sensores, sino también ofrecer una metodología innovadora para la toma de decisiones automatizada basada en inteligencia artificial.

METODOLOGÍA

La metodología a emplear en este proyecto consiste en una combinación de investigación teórica, análisis de tecnologías existentes, simulación y optimización de modelos mediante herramientas de software especializadas.

En la fase inicial, se realizará una revisión de fuentes de referencia reconocidas, como publicaciones científicas y artículos hallados principalmente en el la base de datos del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), debido a su relevancia y alcance internacional en el ámbito de la ingeniería relacionada con el proyecto.

Para la fase de desarrollo y simulación, se utilizarán herramientas de simulación como Simulink en Matlab, lo que permitirá modelar el comportamiento de la red de sensores y evaluar el impacto de los protocolos de comunicación seleccionados. Asimismo, se explorará el uso de Python o de bloques de Deep Learning en Matlab para la implementación de algoritmos de Machine Learning, en función de cuál se adapte mejor a los datos empleados.

El análisis comparativo de los resultados obtenidos permitirá ajustar la configuración de la red y los algoritmos utilizados para optimizar su rendimiento. De esta manera, el proyecto seguirá un proceso iterativo de diseño, prueba y ajuste hasta lograr un sistema con una optimización diseñada de manera eficiente y funcional.

2. ESTADO DE LA CUESTIÓN

INTRODUCCIÓN

Al desarrollar un proyecto de investigación, es esencial analizar los trabajos y soluciones existentes dentro del ámbito de estudio para comprender el estado actual de la tecnología que se busca implementar y la investigación que hay sobre ella. Ante la idea inicial de diseñar una red optimizada de sensores y actuadores en entornos IoT con inteligencia artificial, surgen varias preguntas clave que se buscan responder, como por ejemplo: ¿existen soluciones similares en el mercado? ¿Qué trabajos de investigación han abordado problemáticas similares? ¿Se han obtenido resultados que puedan servir como referencia para este estudio?

Este capítulo tiene como propósito explorar el estado de la cuestión en el ámbito de las redes de sensores y actuadores para aplicaciones IoT a gran escala, con énfasis en los protocolos de comunicación y la integración de técnicas de inteligencia artificial para optimizar su desempeño. Además, se justifica la selección de los protocolos LoRaWAN y MQTT como las tecnologías más adecuadas para este proyecto, argumentando su mejor ajuste al proyecto que se busca realizar en función de estudios previos y características técnicas relevantes.

ESTADO DE LA CUESTIÓN: ANÁLISIS DE PROTOCOLOS IOT

En el campo del Internet de las Cosas, la comunicación eficiente entre sensores y actuadores es fundamental para garantizar un monitoreo y control efectivo de los sistemas distribuidos. Entre los protocolos de comunicación más utilizados en aplicaciones industriales e infraestructuras de gran escala se encuentran LoRaWAN, MQTT, Zigbee y NB-IoT. Diversos estudios han evaluado sus ventajas y desventajas en función de factores como cobertura, consumo energético, escalabilidad y fiabilidad de la transmisión de datos.

2.2.1. COMPARACIÓN DE PROTOCOLOS DE COMUNICACIÓN

Un estudio comparativo sobre tecnologías LPWAN (Low Power Wide Area Network) destaca a LoRaWAN, Sigfox y NB-IoT como los principales protocolos para aplicaciones IoT a gran escala. LoRaWAN se distingue por ofrecer comunicación de largo alcance (hasta 40 km en zonas rurales y 1-5 km en entornos urbanos) con un consumo energético extremadamente bajo, lo que permite que los dispositivos puedan operar durante más de diez años con una única batería [1]. En términos de costo y escalabilidad, LoRaWAN presenta ventajas significativas frente a NB-IoT, que, si bien ofrece una excelente cobertura en

interiores y alta velocidad de transmisión, depende de infraestructuras de redes móviles, lo que incrementa sus costos de implementación y operación.

Por otro lado, Zigbee, aunque es una alternativa eficiente para redes de baja potencia en entornos cerrados o urbanos, tiene un alcance limitado, lo que dificulta su implementación en aplicaciones de monitoreo distribuidas en grandes áreas geográficas. En función de estos criterios, LoRaWAN se presenta como una solución más adecuada para aplicaciones industriales y agrícolas que requieren una conectividad confiable en ubicaciones remotas [2].

2.2.2. APLICACIONES DE LORAWAN Y MQTT EN IOT

El protocolo MQTT (Message Queuing Telemetry Transport), desarrollado por IBM, se ha consolidado como una solución eficiente para la transmisión de datos en entornos IoT con restricciones de ancho de banda. Su arquitectura basada en el modelo publicador-suscriptor permite la comunicación eficiente entre dispositivos y servidores en la nube, facilitando la gestión remota de grandes volúmenes de datos en tiempo real [3].

Un estudio sobre la implementación de LoRaWAN y MQTT en un contexto de monitorización medioambiental en tiempo real demuestra la viabilidad de esta combinación tecnológica en la gestión eficiente de datos de sensores IoT. En dicho trabajo, sensores de temperatura conectados a una placa Arduino transmiten datos mediante un gateway LoRaWAN hacia un servidor MQTT, que posteriormente los redistribuye a un servidor web en la nube para su análisis y visualización [4]. Esta arquitectura facilita la transmisión confiable de datos desde dispositivos IoT hacia sistemas de análisis en la nube, asegurando una baja latencia y un consumo energético optimizado.

En el contexto de aplicaciones industriales, se ha explorado la integración de MQTT con otras tecnologías para mejorar la eficiencia en estaciones de carga de vehículos eléctricos. Los resultados muestran que MQTT es muy eficaz gestionando datos en la nube, pudiendo permitir una supervisión precisa del consumo energético y la optimización de la distribución de recursos en infraestructuras inteligentes [5].

JUSTIFICACIÓN DE LA SELECCIÓN DE LORAWAN Y MQTT

Tras introducir el estado actual de las principales tecnologías estudiadas para el proyecto a realizar, la elección de LoRaWAN y MQTT como protocolos principales para este proyecto se fundamenta en los siguientes aspectos:

- Cobertura y bajo consumo energético: LoRaWAN es la mejor opción para la transmisión de datos en redes de sensores distribuidas en grandes áreas rurales o

industriales, ofreciendo una comunicación eficiente con un consumo energético mínimo [6].

- Fiabilidad y escalabilidad: MQTT permite una comunicación robusta y adaptable entre dispositivos IoT y servidores en la nube, garantizando la entrega de datos en entornos con ancho de banda limitado [3].
- Integración con Inteligencia Artificial: La combinación de LoRaWAN y MQTT proporciona una infraestructura ideal para la implementación de técnicas de aprendizaje automático, permitiendo la optimización dinámica del tráfico de datos y la predicción de eventos en la red de sensores [7].

OPTIMIZACIÓN CON INTELIGENCIA ARTIFICIAL

La incorporación de algoritmos de aprendizaje automático en redes IoT ha demostrado ser una estrategia eficaz para mejorar la toma de decisiones y la eficiencia operativa. Diversos estudios han explorado el uso de técnicas de Machine Learning para optimizar la transmisión de datos en redes IoT, abordando problemáticas como la detección de anomalías, la predicción de fallos y la gestión dinámica de recursos [8].

En este proyecto, se plantea la implementación de modelos de aprendizaje automático para analizar patrones en los datos generados por la red de sensores, permitiendo la identificación de tendencias y la optimización de la transmisión de información y reacción de los actuadores ante la detección de situaciones concretas. Se considerarán distintas técnicas de aprendizaje supervisado y no supervisado, evaluando su desempeño en la mejora de la eficiencia del sistema [7].

CONCLUSIÓN

El análisis del estado de la cuestión ha permitido identificar a LoRaWAN y MQTT como los protocolos de comunicación más adecuados para la implementación de redes de sensores y actuadores en aplicaciones IoT a gran escala. Su combinación ofrece una solución eficiente para la transmisión de datos en entornos industriales y agrícolas, garantizando un bajo consumo energético y una comunicación fiable en tiempo real.

Por otro lado, la integración de inteligencia artificial en la gestión de redes IoT representa una oportunidad para optimizar el tráfico de datos y mejorar la toma de decisiones en este tipo de redes de sensores y actuadores. A partir de todo lo analizado, el siguiente capítulo abordará la justificación más detallada del enfoque propuesto y el diseño de la arquitectura del sistema a desarrollar, buscando justificar la utilidad y aplicación real de este.

3. DEFINICIÓN DEL TRABAJO

JUSTIFICACIÓN

A la vista del capítulo anterior, es necesario realizar un análisis crítico sobre los trabajos previos realizados y establecer claramente por qué se va a desarrollar este proyecto. Aunque es posible apoyar la justificación en una visión filosófica, como la necesidad de crear un vínculo más cercano entre infraestructura y tecnología para lograr mayor sostenibilidad y eficiencia, inicialmente se va a centrar este capítulo en argumentos principalmente técnicos, comerciales y estratégicos.

Tras analizar las tecnologías disponibles y observar la viabilidad de implementar esta red de sensores y actuadores mediante LoRaWan y MQTT, se considera relevante destacar la falta de publicaciones sobre algoritmos de Inteligencia Artificial (IA) o Machine Learning para optimizar este tipo de redes. El crecimiento exponencial del Internet de las Cosas (IoT) ha llevado a la implementación masiva de sensores y actuadores en sectores como la industria, la agricultura y las infraestructuras inteligentes. Sin embargo, la mayoría de las redes IoT actuales operan utilizando modelos reactivos, en los que los dispositivos únicamente responden a eventos en lugar de anticiparse a ellos.

En este contexto, las redes de sensores basadas en LoRaWAN y MQTT ofrecen ventajas clave en términos de cobertura, consumo energético y escalabilidad, pero carecen de sistemas avanzados de optimización y automatización. A pesar de su fiabilidad, sin la integración de técnicas de IA que permitan hacerlas predictivas y autónomas, estas tecnologías podrían quedar rezagadas en comparación con soluciones más avanzadas en automatización.

Este proyecto aborda esta brecha tecnológica mediante la integración de algoritmos de aprendizaje automático para optimizar la gestión de redes de sensores y actuadores, permitiendo un uso más eficiente de los recursos, mejorando la toma de decisiones e incrementando la autonomía de los sistemas IoT.

3.1.1. REFLEXIÓN FILOSÓFICA Y CONTEXTO TECNOLÓGICO

En un mundo plenamente industrializado donde el ahorro, la simplificación y la eficiencia cada vez son más relevantes, aún existen los contratiempos y la complejidad y problemas que estos pueden suponer para los procesos industriales. En la búsqueda de esta mejora constante surgen cuestiones clave: ¿Es posible adelantarse a los imprevistos? ¿Podemos conseguir automatizar los procesos industriales y darles la capacidad de aprender, adaptarse y tomar decisiones inteligentes con los menores gastos asociados posibles?

Esta cuestión nos obliga a pensar en la relación entre los datos que generan los sistemas físicos y todo el proceso asociado que conlleva tomar decisiones a partir de ellos en estos entornos, no sólo el papel humano, sino también toda la energía, infraestructura e inversión asociada a decisiones que, tomadas con anterioridad, pueden reducir enormemente los gastos y optimizar el funcionamiento de muchos procesos. Afortunadamente, la tecnología actual nos permite plantearnos una vía alternativa: la automatización inteligente.

Hoy, el procesamiento de datos y los algoritmos de Inteligencia Artificial ocupan un papel central en la evolución de muchos sectores clave. Esto se debe a que permiten pasar de una lógica que tan solo reacciona a una lógica predictiva y optimizada. Por ejemplo, en una planta fotovoltaica, detectar con antelación una leve desviación en la inclinación de un panel puede evitar una pérdida significativa de rendimiento energético. Si esa corrección se automatiza, no solo se mejora la eficiencia energética, sino que se eliminan tiempos de inactividad y se reducen los costes operativos.

Este proyecto nace de esa lógica que consiste en aprovechar el potencial de la IA para optimizar redes de sensores y actuadores, de forma que no solo recopilen datos, sino que interpreten, anticipen y actúen. La integración de sensores, redes de comunicación eficientes y modelos de IA no representa solo un avance tecnológico, sino una oportunidad real para redefinir la gestión de recursos en entornos industriales mediante sistemas más adaptativos, precisos y sostenibles.

Especialmente en sectores como las energías renovables, donde la producción depende de variables cambiantes y dinámicas, automatizar la interpretación de datos en tiempo real es clave para conseguir soluciones eficientes y robustas.

.

3.1.2. JUSTIFICACIÓN TÉCNICA Y DE MERCADO

La propuesta técnica de este proyecto se fundamenta en la necesidad de construir sistemas de comunicación IoT más eficientes, sostenibles y capaces de escalar en entornos industriales reales. Esta necesidad se complementa con una oportunidad clara de mercado: aprovechar tecnologías ya existentes, como son LoRaWAN y MQTT, y combinarlas con Inteligencia Artificial para ofrecer una solución de automatización inteligente. A continuación, se detallan los aspectos técnicos y comerciales que justifican el desarrollo de este sistema.

3.1.2.1. Razones Técnicas

- Cobertura y bajo consumo

El uso de tecnologías de red de baja potencia y largo alcance, como LoRaWAN, permite desplegar soluciones en entornos industriales o agrícolas donde otras tecnologías resultan inviables por su limitado alcance o elevado consumo energético. LoRaWAN es

especialmente adecuada para sensores distribuidos en grandes superficies, ya que permite comunicaciones a varios kilómetros con un consumo mínimo, lo que resulta esencial en entornos donde no se dispone de acceso constante a la red eléctrica. Esta tecnología, basada en modulación de espectro ensanchado, es capaz de mantener transmisiones superiores a 10 km en zonas rurales, con dispositivos que pueden funcionar durante años sin reemplazo de batería lo cual se traduce en una mayor autonomía de los dispositivos y menor necesidad de mantenimiento [9].

- Fiabilidad y escalabilidad

Por un lado, a diferencia de otras tecnologías, LoRaWAN permite la comunicación bidireccional y soporta múltiples clases de dispositivos, lo que permite adaptarse a diferentes necesidades de latencia y frecuencia de transmisión [9]. Por otro lado, el sistema basado en gateways y servidores centrales permite escalar el número de sensores sin comprometer la estabilidad del sistema. Además, protocolos como MQTT aseguran una comunicación eficiente y robusta gracias a su bajo peso, simplicidad y capacidad de operación en redes inestables. Su modelo publish/subscribe facilita la gestión de miles de nodos y su integración con plataformas de análisis y toma de decisiones en la nube [10].

- Integración con Inteligencia Artificial

La arquitectura propuesta permite aplicar algoritmos de Machine Learning tanto en la nube como en otros entornos. Esto habilita funcionalidades avanzadas como la detección anticipada de anomalías o eventos, la optimización del consumo energético o el control predictivo de actuadores. Se han demostrado beneficios tangibles en sistemas agrícolas y de automatización industrial al incorporar modelos inteligentes, desde el ajuste automático de riego hasta la predicción de ataques a nivel de red [10].

3.1.2.2. Justificación de mercado

- Solución eficiente

En el contexto de redes industriales y energéticas, donde los sistemas deben operar de forma continua y distribuida, tecnologías como LoRaWAN y MQTT representan una alternativa estratégica frente a soluciones no accesibles al público. Su naturaleza abierta elimina costes de licencia y permite implementar arquitecturas flexibles que se adaptan a diferentes escalas de implementación. Además, al optimizar el consumo energético y reducir el tráfico de red innecesario, estas tecnologías contribuyen a prolongar la vida útil de los nodos y a disminuir los requerimientos de mantenimiento, lo que se traduce en una operación más sostenible y rentable a largo plazo [9]

- Uso creciente de IoT en infraestructuras inteligentes

En los últimos años, la digitalización de infraestructuras industriales, energéticas y urbanas ha impulsado una adopción acelerada de tecnologías IoT como parte fundamental de sistemas de control y supervisión. Esta tendencia responde a la necesidad de mejorar la

eficiencia operativa, reducir los costes de mantenimiento y habilitar una toma de decisiones basada en datos en tiempo real.

La creciente presencia de sensores conectados en instalaciones como plantas solares, redes de distribución eléctrica, estaciones meteorológicas o sistemas de monitorización estructural es una muestra clara de esta transformación. En estos contextos, los dispositivos IoT ya no se consideran una tecnología experimental, sino una herramienta clave para mejorar la sostenibilidad, la seguridad y la resiliencia operativa

3.1.3. PROPUESTA DE VALOR

Esta propuesta busca ser más que solo una hipótesis sobre la posible optimización de una red de sensores. Una vez desarrollado, este proyecto puede convertirse en una arquitectura modular, fácilmente replicable y con un coste bajo, además de proporcionar múltiples ventajas en el ámbito industrial. Entre ellas, algunos aspectos clave que haría posible este proyecto son los que aparecen a continuación.

- Reducir el consumo energético y los costes de mantenimiento, al limitar el envío de datos exclusivamente a eventos anómalos y señales periódicas de estado. Esto permite que los sensores operen durante largos periodos sin intervención, disminuyendo la necesidad de recargas o sustituciones y reduciendo el número de visitas técnicas a campo.
- Prevenir fallos antes de que ocurran, gracias a la implementación de algoritmos de inteligencia artificial que analizan los patrones históricos de comportamiento del sistema. Esta detección proactiva permite intervenir con antelación, evitando paradas imprevistas o daños en los equipos.
- Trasladarse fácilmente a otras instalaciones sin rediseño completo, ya que el sistema ha sido diseñado sobre protocolos estándar (MQTT y LoRaWAN), y su lógica de funcionamiento es adaptable a distintos entornos mediante una simple reconfiguración. Esto permite desplegar nuevos nodos o replicar la arquitectura en otras instalaciones industriales con mínima intervención técnica.

Desde el punto de vista comercial, la solución es altamente destacable por varios motivos justificados en los siguientes puntos.

- Llena un vacío real en el mercado de automatización industrial, donde muchas infraestructuras aún dependen de sistemas cableados, propietarios o poco flexibles, que dificultan su actualización o escalado. Esta propuesta, en cambio, ofrece una vía práctica para avanzar hacia entornos inteligentes con bajo coste de entrada.
- Tiene alto potencial de rentabilidad, al reducir costes operativos y aumentar la eficiencia energética, al reducir drásticamente el volumen de datos transmitidos, optimizar el funcionamiento de los actuadores y disminuir el consumo energético.

Estos beneficios se traducen en menores costes operativos y una mayor durabilidad de los equipos.

- Está alineada con los Objetivos de Desarrollo Sostenible, en particular con los relacionados con la industria, la innovación y la acción climática. Al favorecer una operación más limpia, autónoma y basada en datos, esta propuesta contribuye a una gestión responsable de los recursos tecnológicos y energéticos. En definitiva, es una propuesta pensada para funcionar en la realidad. Una herramienta concreta para anticiparse, optimizar y evolucionar.

En definitiva, es una propuesta pensada para funcionar en la realidad. Una herramienta concreta para anticiparse, optimizar y evolucionar, con un enfoque técnico y orientado a la aplicabilidad directa.

OBJETIVOS

3.2.1. OBJETIVO GENERAL

Diseñar una red de sensores basada en tecnología IoT capaz de supervisar y mejorar el rendimiento de sistemas industriales distribuidos, mediante la combinación de protocolos de comunicación de bajo consumo y técnicas de inteligencia artificial orientadas a la toma de decisiones en tiempo real. Para ello, se realizará un análisis exhaustivo de los principales protocolos utilizados en este tipo de aplicaciones. Como resultado, se ha determinado que LoRaWAN y MQTT constituyen la solución más adecuada para garantizar una transmisión eficiente y escalable. En capítulos posteriores, se abordará con detalle el modelo de IA implementado y la metodología empleada para su entrenamiento y validación.

3.2.2. OBJETIVOS ESPECÍFICOS

- Analizar el comportamiento del sensor ADXL345 en condiciones industriales, evaluando su capacidad para detectar inclinaciones, vibraciones y desplazamientos relevantes para el control del sistema.
- Describir la arquitectura de la red de comunicaciones y el flujo de datos entre nodos, incluyendo la implementación de LoRaWAN para el enlace de largo alcance y MQTT como protocolo de transporte hacia el entorno de análisis en la nube.
- Desarrollar un modelo de Inteligencia Artificial alojado en la nube que permita reconocer patrones relevantes en los datos recogidos por los sensores y generar decisiones optimizadas en función del contexto.

- Evaluar la lógica de respuesta de los actuadores a partir de las salidas del modelo de IA, incluyendo la generación de señales de control como valores PWM.
- Implementar una simulación del sistema en Simulink de Matlab, diferenciando claramente los bloques funcionales correspondientes a la transmisión de datos y al procesamiento mediante IA, con el fin de representar de forma estructurada la lógica del sistema y evaluar su comportamiento de forma segmentada y controlada.
- Estudiar la aplicabilidad del sistema en una instalación real, considerando las condiciones de operación, viabilidad técnica y beneficios potenciales en términos de eficiencia, mantenimiento y adaptabilidad.

METODOLOGÍA

El desarrollo del proyecto se ha estructurado en cuatro fases complementarias, cada una centrada en un objetivo específico que permite avanzar de forma progresiva y coherente hacia la implementación completa del sistema propuesto. A continuación, se detallan los métodos y herramientas utilizados en cada una de estas etapas.

- Revisión bibliográfica y análisis técnico: El proceso comenzó con una revisión exhaustiva de literatura científica y documentación técnica. Esta investigación permitió identificar las tecnologías más adecuadas y establecer los requisitos funcionales del sistema. Esta labor se ha mantenido activa durante todo el desarrollo, sirviendo para fundamentar decisiones específicas del diseño, validar metodologías implementadas y reforzar las conclusiones obtenidas en las fases posteriores. Para ello se recurrió a fuentes académicas lo más actuales posible, especialmente artículos del IEEE y otras bases de datos de referencia.
- Diseño del sistema en entorno de simulación: Con los fundamentos técnicos establecidos, se ha procedido a la representación de un sistema equivalente en Simulink de Matlab. En esta fase se construyó una arquitectura modular que distingue dos bloques funcionales claramente separados:
  - La lógica de transmisión y comunicación de datos
  - El procesamiento inteligente, donde se simula la actuación del modelo de IA a partir de los datos recibidos.

  Esta segmentación permite una evaluación independiente del comportamiento de cada componente del sistema y facilita futuras adaptaciones o ampliaciones.
- Implementación del modelo de inteligencia artificial: El desarrollo del algoritmo de optimización se llevó a cabo en Python. A partir de los datos sensorizados, se entrenaron modelos de Machine Learning con el objetivo de detectar eventos significativos y generar salidas adaptativas. Se exploraron enfoques tanto supervisados como no supervisados, y se compararon distintas configuraciones para optimizar el rendimiento del sistema.

- Evaluación del sistema propuesto: Durante la fase de evaluación se analizaron los resultados obtenidos a través de las simulaciones realizadas en Simulink y de los modelos desarrollados en Python, con el objetivo de validar el comportamiento del sistema y extraer conclusiones fundamentadas. Se han utilizado representaciones visuales, métricas estadísticas y gráficos comparativos para examinar la coherencia entre las predicciones del modelo de inteligencia artificial y las salidas simuladas, así como para verificar la eficacia del procesamiento inteligente de eventos. Esta combinación de análisis técnico y visual ha permitido poder comprobar el cumplimiento de los objetivos planteados y detectar oportunidades de mejora y de trabajos futuros.

PLANIFICACIÓN Y ESTIMACIÓN ECONÓMICA

A continuación, se presenta la planificación temporal y la estimación económica del sistema propuesto, considerando su implementación en una planta fotovoltaica estándar. Aunque algunos elementos podrían variar en función del sector o del entorno, se considera que esta estimación puede extrapolarse a otros procesos industriales con adaptaciones mínimas.

3.4.1. FUNCIONAMIENTO DEL SISTEMA

Antes de detallar los costes y la configuración técnica, es importante introducir brevemente el contexto operativo del sistema propuesto para facilitar la comprensión a un lector no especializado. Este sistema se basa en una red distribuida de sensores ADXL345, un tipo de acelerómetro digital de bajo consumo que permite medir inclinaciones y vibraciones en estructuras en los tres ejes. Aunque más adelante se explicará en profundidad su funcionamiento y especificaciones, cabe destacar que su integración en entornos industriales permite detectar anomalías estructurales o cambios en la orientación de los equipos monitorizados.

El sistema utiliza tecnología LoRaWAN para la transmisión de los datos desde los sensores a gateways intermedios, y posteriormente emplea el protocolo MQTT para comunicar los datos procesados hacia un servidor en la nube. Este servidor se encarga del análisis de los datos y de la toma de decisiones en tiempo real. En esta arquitectura, la eficiencia energética y la escalabilidad son prioritarias. Por ello, los sensores están programados para transmitir únicamente en dos casos: ante la detección de un evento anómalo o cada 60 segundos mediante un mensaje de tipo keepalive. Esto permite minimizar el tráfico de red y reducir significativamente el consumo de energía.

En cuanto al tamaño del mensaje transmitido, se ha definido un payload optimizado de 7 bytes, que permite mantener una precisión suficiente para la tarea de monitorización sin superar los límites técnicos impuestos por LoRaWAN.

3.4.2. DIMENSIONAMIENTO DE LA RED DE COMUNICACIONES

La configuración de red propuesta contempla 1.000 sensores ADXL345 y 20 gateways LoRaWAN, es decir, una media de 50 sensores por gateway. Esta decisión se basa tanto en una investigación bibliográfica como en un análisis realista a partir de los datos obtenidos en una simulación real, en la que se modeló el comportamiento del microcontrolador que regula cuándo se transmite la información, y partiendo del supuesto de que el sistema se implantaría en una planta fotovoltaica de gran tamaño. Esta planta se estima que sea mayor o igual a 50 MW, ya que esto equivale a una planta con aproximadamente 1000 trackers y lo habitual es establecer un sensor por tracker. Uno de los objetivos de este trabajo ha sido precisamente estudiar la viabilidad de desplegar una red IoT a gran escala en este tipo de infraestructuras, por lo que la selección de esta magnitud de red es un aspecto clave.

Tal como se ha introducido, en el sistema diseñado, un sensor solo envía datos en dos situaciones: cuando se detecta una inclinación anómala (evento), o de forma periódica una vez cada minuto (como keepalive). Fuera de estos casos, no se transmite nada, lo que reduce el consumo y el tráfico de datos. A partir de los datos registrados durante más de 55 horas de simulación y analizando más de 2 millones de muestras, se identificaron 7.899 eventos únicos. Esto equivale a un valor estimado de 143 eventos por hora por sensor (7899 eventos únicos divididos entre 55.1 horas de simulación). Si se suman los 60 mensajes keepalive por hora, cada sensor transmite en promedio 203 mensajes por hora. Es importante matizar que este valor estimado sería correspondiente a un escenario de carga máxima registrado durante la simulación, en el que los eventos eran frecuentes debido a un umbral bajo de detección de anomalías para estudiar cómo responde el sistema en escenarios donde se requiere máxima precisión o anticipación ya que es útil para validar el rendimiento de la red y del modelo de IA en condiciones extremas o de alta sensibilidad. En condiciones reales, el número de eventos sería notablemente inferior, y por tanto, los mensajes keepalive recuperarían protagonismo. Aun así, esta estimación ha sido útil para garantizar la viabilidad del sistema bajo condiciones exigentes.

Cada mensaje, de 7 bytes de datos reales más algunos bytes de cabecera del protocolo, ocupa un tiempo en el aire o ToA (duración de la transmisión) de aproximadamente 56 milisegundos si se utiliza la configuración más eficiente (SF7 y 125 kHz). Por tanto, cada sensor consume unos 11.37 segundos por hora de transmisión. Multiplicando por 50 sensores por gateway, se llega a unos 568.5 segundos por hora, lo que representa un 15.8% del tiempo disponible para transmitir [11]. Este valor se distribuye entre los 8 canales disponibles en el chip de los gateways, reduciendo el uso real por canal a menos del 2%, dentro de los límites establecidos por la normativa europea para esta banda de frecuencias [12], [13].

Además de cumplir la normativa, esta configuración permite redundancia: cada sensor puede ser escuchado por más de un gateway, lo que mejora la fiabilidad del sistema ante posibles interferencias o pérdidas. Por ello, aunque teóricamente un gateway podría soportar hasta 300 sensores, se ha optado por una solución más conservadora y robusta con 50 sensores por Gateway [13]. Esta decisión está respaldada por estudios recientes, recomendaciones del estándar IEC 62933-5-1 [14], y datos de despliegues reales en plantas de más de 50 MW,

donde se ha documentado que configuraciones similares con 1.000 sensores y 20 gateways proporcionan una cobertura granular adecuada (1 sensor por cada 0.2 hectáreas aproximadamente) y un ratio de entrega de paquetes superior al 99.8% [15], [16].

En cuanto a la arquitectura en la nube, los mensajes enviados por los sensores se transmiten a través de los gateways al servidor LoRaWAN, que actúa como puente hacia un sistema de mensajería MQTT. Este sistema publica los datos para que sean procesados por los algoritmos de Inteligencia Artificial en tiempo real. A pesar de manejar hasta 56 mensajes por segundo en total, la carga está muy por debajo de los límites técnicos de los brokers MQTT modernos, lo que garantiza una operación estable [17], [18].

En cuanto a los temas mencionados como el funcionamiento concreto de los sensores, la frecuencia de transmisión, el tamaño de los mensajes y el procesamiento de los datos, se analizarán en detalle en el siguiente capítulo, donde se presentarán las fórmulas, resultados de simulación y decisiones técnicas explicadas.

3.4.3. ESTIMACIÓN ECONÓMICA TOTAL

Esta sección presenta la estimación de costes necesarios para desplegar el sistema propuesto en una planta fotovoltaica de referencia. Podemos verlo en la tabla a continuación [Tabla 1], donde se han considerado los gastos asociados a hardware, comunicaciones, desarrollo, fabricación, montaje y puesta en marcha. La estimación se basa en el dimensionamiento explicado en el apartado anterior (1000 sensores y 20 gateways), así como en supuestos realistas derivados de experiencias industriales y documentación técnica.

Cabe destacar que en el coste de hardware no se incluye el precio de los actuadores, ya que en el planteamiento del sistema se considera que la inteligencia artificial proporciona una señal de consigna al sistema de control de actuadores ya existente. Normalmente, estos sistemas de control son independientes y ya están integrados en los procesos industriales, por lo que no es necesario incorporar ni valorar su coste en esta estimación.

Concepto	**Coste Unitario (€)**	**Cantidad**	**Coste Total (€)**
Hardware (por unidad)			
Encapsulado	2.00	1000	2,000.00
Microcontrolador	3.00	1000	3,000.00
Alimentación	5.00	1000	5,000.00
Sensor	1.80	1000	1,800.00
Total Hardware	11.80	1000	11,800.00
Red de Comunicaciones			
Gateways	300.00	20	6,000.00

Total Comunicaciones			6,000.00
Desarrollo (I+D)			7,200.00
Fabricación y Montaje			
Acopio (2 semanas)			720.00
Fabricación (1 mes)			1,500.00
Programación y pruebas (2 semanas)			9,600.00
Instalación y montaje (1 semana)			6,600.00
Puesta en Marcha (PeM) (1 semana)			4,800.00
Total Fabricación y Montaje			23,220.00
Total General (€)			**48,220.00**

Tabla 1: Estimación de gastos totales

3.4.4. PLANIFICACIÓN TEMPORAL

Desde el punto de vista temporal, la Tabla 2 a continuación sintetiza las diferentes fases de implementación, incluyendo duración y coste asociado. Se aprecia que las tareas más intensivas en tiempo y recursos están ligadas a programación, pruebas, e instalación del sistema completo. Estas fases son críticas, ya que implican tanto la validación técnica del sistema como su puesta en funcionamiento en el entorno real. Una planificación precisa en estos puntos permite optimizar los recursos disponibles, minimizar riesgos y asegurar la correcta integración del sistema con la infraestructura existente.

Fase	**Duración**	**Coste (€)**
Acopio de materiales	2 semanas	720.00
Fabricación	1 mes	1,500.00
Programación y pruebas	2 semanas	9,600.00
Instalación y montaje	1 semana	6,600.00
Puesta en Marcha (PeM)	1 semana	4,800.00
Total	**10 semanas**	**23,220.00**

Tabla 2: Planificación económica del proyecto

4. SISTEMA PLANTEADO

La red de sensores física, tal como se ha ido introduciendo a lo largo del trabajo, se ha planteado para estar dividida en tres partes clave: el funcionamiento del sensor, la transmisión de los datos y su procesamiento con los algoritmos de Inteligencia Artificial. A pesar de que se han hecho adaptaciones de cara a la simulación propia de esta red, que se explicarán en los siguientes capítulos, es relevante para el trabajo explicar el funcionamiento real tanto del sensor como de los dos protocolos estudiados (LoRaWan y MQTT) así como el funcionamiento completo del flujo de la red, para así entender las modificaciones planteadas más adelante desde un marco técnico y justificado en la realidad.

FUNCIONAMIENTO DEL SENSOR

El sistema parte del uso del sensor ADXL345, un acelerómetro triaxial digital ampliamente utilizado en aplicaciones industriales por su bajo consumo y alta precisión. Este sensor es capaz de medir aceleraciones en los ejes X, Y y Z, permitiendo detectar tanto aceleración dinámica (como vibraciones o impactos) como aceleración estática (gravedad). Según la orientación del dispositivo frente a los ejes, cuya referencia de medida podemos observarla en la Figura 2. En función de la orientación del dispositivo respecto a la dirección de la gravedad, el sensor entregará valores característicos en cada eje. Por ejemplo, cuando la cara superior del sensor apunta directamente hacia abajo, se registra una aceleración de -1g en el eje Z y 0g en los ejes X e Y, como podemos analizar en la Figura 3 [19].

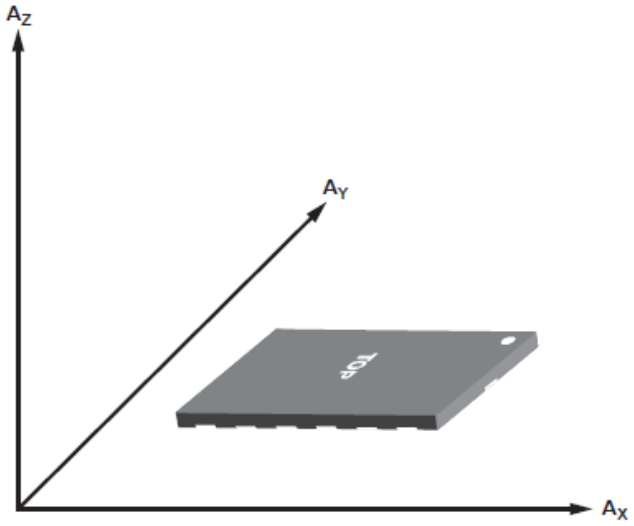


Figura 2: Ejemplo de eje del sensor del acelerómetro ADXL345 [20]

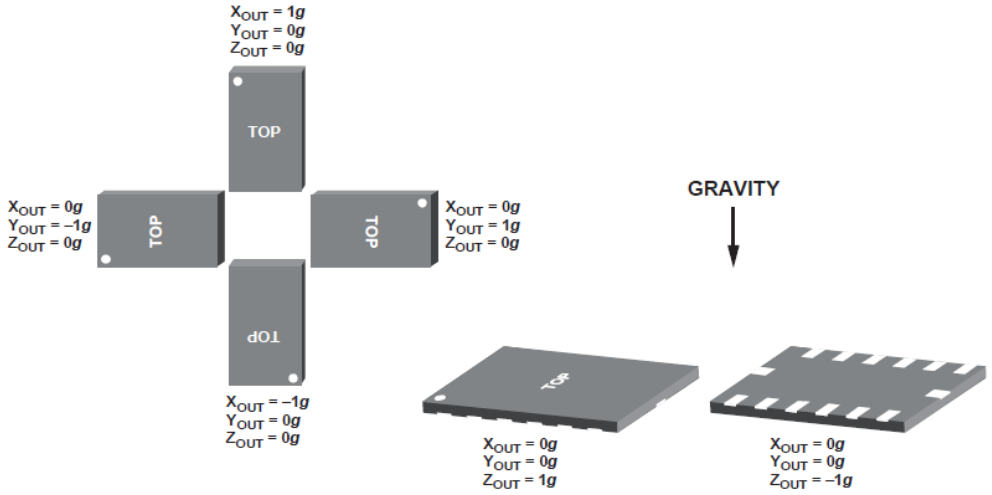


Figura 3: Respuesta de salida ADXL345 a la orientación de la gravedad [20]

En el contexto de una planta fotovoltaica, estos sensores pueden colocarse sobre las estructuras que sostienen los paneles solares, de forma que monitoricen su inclinación respecto al suelo. Por ejemplo, si una estructura pierde estabilidad, se ve afectada por el viento o se desplaza por asentamiento del terreno, el sensor detectará un cambio en las componentes de aceleración y podrá alertar al sistema para que se realice un ajuste o una revisión.

Esta capacidad de medición por ejes es fundamental para calcular la inclinación de la superficie en la que está montado el sensor. A partir de las aceleraciones registradas en los tres ejes (a_x, a_y, a_z), el microcontrolador calcula las rotaciones respecto al eje X y al eje Y, siguiendo las 1,2 y 3 mostradas a continuación.

$$rot_x = tan^{-1}\left(\frac{a_x}{\sqrt{a_x{}^2+a_y{}^2}}\right) * \frac{180º}{\pi} \qquad (1)$$

$$rot_y = tan^{-1}\left(\frac{a_y}{\sqrt{a_x{}^2+a_y{}^2}}\right) * \frac{180º}{\pi} \qquad (2)$$

$$grav = \sqrt{a_x{}^2 + a_y{}^2 + a_z{}^2} \qquad (3)$$

Estas conversiones permiten expresar las inclinaciones rot_x y rot_y en grados, lo que facilita la interpretación directa del ángulo de desviación respecto a la horizontal. La multiplicación por 180/π se realiza para convertir el resultado de la función arcotangente (que se obtiene en radianes) a grados, una unidad más útil en aplicaciones prácticas. Por su parte, grav representa el módulo de la aceleración total en m/s^2, útil para detectar impactos o pérdidas de estabilidad [19], [21].

La información que genera el sensor no se transmite directamente, sino que es primero recibida y procesada por un microcontrolador ESP32. El microcontrolador es responsable de ejecutar la lógica de control de eventos y decidir si se debe realizar una transmisión [22]. Esta decisión está basada en umbrales definidos para las variables rot_x, rot_y y grav, que permiten distinguir entre situaciones normales y potenciales eventos anómalos. Además, si durante un periodo continuo de 60 segundos no se ha detectado ningún evento, el microcontrolador enviará automáticamente un mensaje tipo keepalive para confirmar que el nodo sigue operativo.

En el sistema planteado, cada microcontrolador está vinculado a un único sensor, lo que facilita un control preciso e individualizado. Los datos enviados por el sensor son las aceleraciones en los tres ejes, las cuales se emplean en el microcontrolador para calcular las variables físicas de interés. Este diseño permite mantener una arquitectura modular y escalable, preparada para integrar técnicas de inteligencia artificial que optimicen el comportamiento de la red a partir de los datos recogidos.

PLANTEAMIENTO RED DE COMUNICACIONES

El sistema propuesto se apoya en una arquitectura de red distribuida basada en tecnologías de comunicación de bajo consumo y largo alcance, específicamente LoRaWAN y MQTT. La justificación de la selección de estas tecnologías ya ha sido explicado y razonado y se ha justificado su idoneidad para aplicaciones industriales que requieren escalabilidad, eficiencia energética y una comunicación en tiempo real.

Esta red de comunicaciones no solo cumple una función de transporte de datos, sino que es un punto clave en la toma de decisiones automatizada: conecta los sensores con el sistema de Inteligencia Artificial en la nube y con los actuadores que ejecutan las respuestas correctivas. En las siguientes secciones se explica en más detalle el funcionamiento de cada uno de los dos protocolos utilizados y el flujo completo de información a través de la red, lo que permite comprender cómo se establece una comunicación efectiva y en tiempo real entre el entorno físico y el procesamiento digital.

4.2.1. FUNCIONAMIENTO LORAWAN

LoRaWAN (Long Range Wide Area Network), tal como se ha introducido, es un protocolo de comunicación diseñado para redes de bajo consumo y largo alcance, características especialmente útiles en entornos industriales como plantas fotovoltaicas, especialmente al tener en cuenta el tamaño objetivo planteado. En el sistema propuesto, este protocolo permite la transmisión de datos desde los nodos sensores hasta la nube a través de gateways, sin necesidad de infraestructura cableada.

En la red planteada, el sensor está conectado a un microcontrolador, que a su vez está conectado a un módulo LoRa que se encarga de encapsular los datos recogidos y enviarlos de forma inalámbrica. Estos datos se transmiten únicamente cuando se detecta un evento anómalo en la inclinación del panel o cuando transcurren 60 segundos sin transmisión, activando un mensaje tipo keepalive. Esta lógica de envío controlado, programada en el microcontrolador, permite reducir significativamente el tráfico en la red y prolongando la vida útil de los nodos [23].

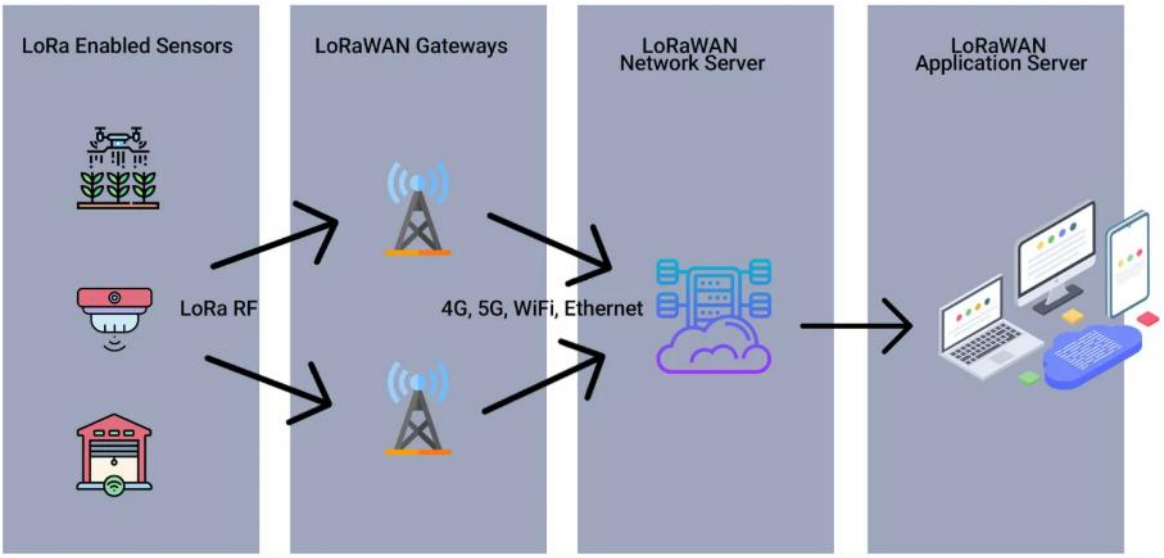


Figura 4: Flujo de funcionamiento de LoRaWan [24]

Tal como aparece explicado de manera más visual en la Figura 4, cada paquete de datos es enviado por el microcontrolador y es recibido por un gateway LoRaWAN, que actúa como punto de acceso entre la red de sensores y la infraestructura de red IP (Internet Protocol). Estos gateways están conectados a un servidor de red LoRaWAN, responsable de gestionar la autenticación, la integridad y el desencapsulado de los paquetes recibidos [25]. Una vez validados, los datos se reenvían al broker MQTT, donde continúan su procesamiento y análisis. Esta estructura modular y jerárquica permite que el sistema pueda escalarse fácilmente a cientos o miles de nodos sin saturar la red ni comprometer la fiabilidad del sistema [23].

En la siguiente sección se explicará con más detalle el funcionamiento del protocolo MQTT y cómo se integra con los datos recibidos desde LoRaWAN para alimentar los algoritmos de inteligencia artificial desplegados en la nube.

4.2.2. FUNCIONAMIENTO MQTT

El protocolo MQTT (Message Queuing Telemetry Transport) es un estándar ligero y eficiente para la transmisión de datos en redes IoT. Está diseñado bajo un modelo de comunicación publish-subscribe (publicar-suscribirse), lo que permite una transmisión asincrónica de mensajes entre dispositivos. Este modelo resulta especialmente útil en aplicaciones como la desarrollada en este trabajo, donde múltiples sensores deben enviar información a un sistema central de análisis y respuesta [26].

En el sistema planteado, el servidor de red LoRaWAN actúa como cliente MQTT y publica los datos extraídos del sensor en un broker MQTT. Este broker, que puede encontrarse en un servidor en la nube, es el nodo central que recibe las publicaciones y las redistribuye a todos los clientes que estén suscritos al canal correspondiente (el cual se llama topic).

En la red planteada en el trabajo, entre los clientes suscritos se encuentra un sistema de IA que analiza los datos recibidos en tiempo real. Este sistema procesa los valores de inclinación y aceleración, y en función de sus resultados, genera una respuesta para corregir la posición del sensor, concretamente un nuevo valor de PWM que debe ser enviado de vuelta al microcontrolador correspondiente para ser comunicado al actuador, que es un servo.

La publicación de esta respuesta se realiza también a través del broker MQTT, pero en un canal de comunicación descendente (llamado downlink). El servidor LoRaWAN, al estar suscrito también a este canal, recupera el mensaje y prepara un nuevo paquete LoRaWAN que será transmitido en el flujo explicado. Este proceso cierra el ciclo completo de respuesta del sistema.

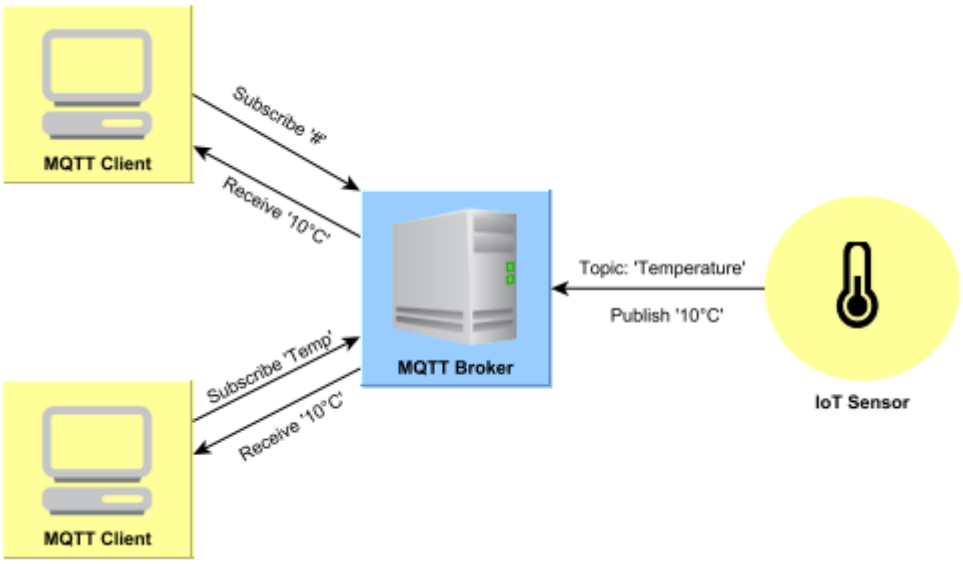


Figura 5: Patrón de Publish-Subscribe en MQTT [17]

Para poder entenderlo de manera visual y simplificar el funcionamiento de MQTT, puede resultar útil observar la Figura 5, donde se puede ver que el funcionamiento de este protocolo se basa en un esquema donde un sensor (como el IoT Sensor de temperatura mostrado) publica un mensaje en un topic determinado (por ejemplo, Temperatura), el cual es gestionado por el broker MQTT. Todos los clientes que estén suscritos a ese topic recibirán el mensaje, facilitando una arquitectura descentralizada y eficiente. Esta lógica se refleja en el sistema desarrollado, donde los datos del sensor no se publican directamente desde el nodo sensor, sino que son primero procesados por el microcontrolador y transmitidos mediante LoRaWAN a un gateway. Desde ahí, los datos son enviados a un servidor LoRaWAN, que actúa como cliente MQTT y publica la información en el broker correspondiente, continuando después con el flujo ya explicado.

4.2.3. FLUJO COMPLETO DE LA RED

La red diseñada se articula en torno a una comunicación estructurada, que conecta sensores físicos con un sistema de análisis inteligente en la nube. Esto permite combinar la eficiencia de LoRaWAN para la transmisión de datos a larga distancia con la flexibilidad del protocolo MQTT para la gestión de mensajes en tiempo real.

Como se observa en la Figura 6, que representa el funcionamiento de la red de manera simplificada, esta arquitectura modular facilita la escalabilidad del sistema, es decir, nuevos sensores pueden integrarse sin alterar el comportamiento del resto de nodos. La separación entre capas proporciona flexibilidad, seguridad y facilidad de mantenimiento.

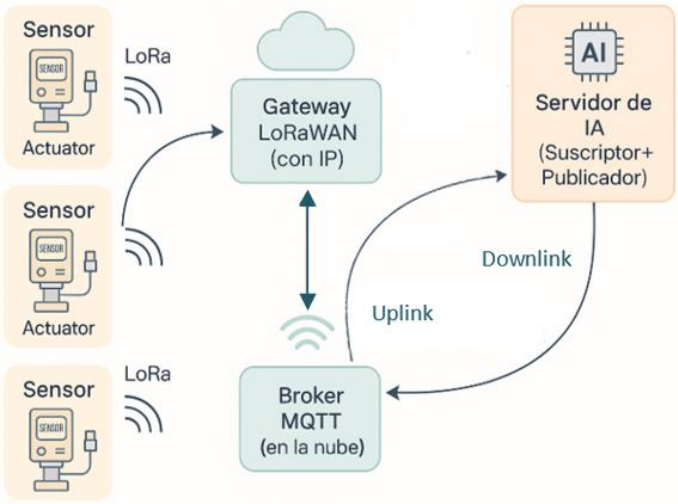
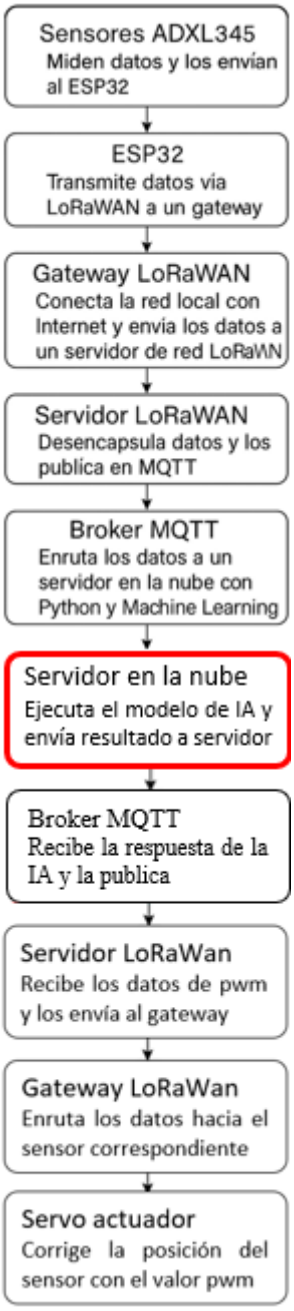


Figura 6: Funcionamiento Red de Comunicaciones



Tal como se representa en la Figura 7, el proceso comienza con los sensores ADXL345, que recogen información sobre la inclinación de los paneles solares. Estos datos son enviados a un microcontrolador ESP32, que calcula variables clave (rot_x, rot_y y grav) y decide si debe transmitir. En caso afirmativo, los datos se encapsulan mediante LoRaWAN y se envían a un gateway, que los redirige hacia un servidor LoRaWAN. Este servidor desencapsula la información y la publica en un broker MQTT.

Desde el broker, los datos son dirigidos hacia un servidor en la nube que contiene el modelo de inteligencia artificial. Este modelo analiza los valores recibidos y determina una respuesta, como un valor PWM para corregir la posición de un actuador. El proceso hasta aquí es parte del uplink y a partir de ahora se determina downlink, algo que se idenfica fácilmente si nos fijamos en el bloque rojo de la Figura 7. A continuación, la respuesta es publicada nuevamente en el broker, recibida por el servidor LoRaWAN y transmitida al microcontrolador del sensor correspondiente, completando el ciclo.

Una parte clave del proceso y que no se ha tratado en mucha profundidad es el funcionamiento del actuador. Aunque su control directo no forma parte del microcontrolador, sí recibe una señal de consigna (un valor PWM) que indica la acción que debe realizar. Esta señal se deriva del valor calculado por el modelo de IA y es interpretada por el sistema de control del actuador, que normalmente opera de forma independiente.

El uso de LoRaWAN permite asegurar la conectividad en grandes superficies como una planta fotovoltaica, donde es inviable una infraestructura cableada y se requiere un consumo energético mínimo. Por su parte, MQTT permite desacoplar la lógica de publicación y recepción, lo que facilita la integración de algoritmos inteligentes en la nube sin necesidad de conexiones permanentes.

Figura 7: Flujo completo de la red

En el siguiente apartado se profundizará en cómo el modelo de IA implementado permite optimizar tanto el comportamiento de los actuadores como la eficiencia de la red de comunicaciones, reduciendo el tráfico innecesario y anticipando posibles fallos.

JUSTIFICACIÓN DE LA OPTIMIZACIÓN DEL SISTEMA

Es importante tener en cuenta que la optimización del sistema no es un complemento añadido del trabajo desarrollado, sino el núcleo central de este. El propio título del proyecto lo refleja y desde el principio el objetivo principal ha sido diseñar una red IoT eficiente para plantas fotovoltaicas y demostrar cómo esta puede ser optimizada mediante IA. Esta optimización se ha estructurado en dos fases diferenciadas pero complementarias. Por un lado, la reducción del número de transmisiones mediante algoritmos de aprendizaje no supervisado, y por otro la predicción de la respuesta óptima del sistema ante cada evento, lo que permite un ahorro energético significativo y una corrección más precisa de las desviaciones detectadas.

4.3.1. REDUCCIÓN DE TRANSMISIONES MEDIANTE APRENDIZAJE NO SUPERVISADO

En la primera fase, se ha buscado minimizar el tráfico de red sin comprometer la capacidad de detección de eventos relevantes. Para ello, se ha analizado el comportamiento de las variables rot_x, rot_y y grav mediante técnicas de aprendizaje no supervisado, en las cuales se hablará en más detalle en los siguientes capítulos. Estas técnicas han permitido identificar patrones de comportamiento que se salían de lo normal y así detectar anomalías, pudiendo identificar umbrales físicos que categorizan los eventos que pueden suponer un riesgo para la correcta colocación de las placas solares, en este caso. Sin embargo, este enfoque hace el sistema más autónomo y es adaptable a distintos entornos industriales.

Gracias a este análisis se consigue el funcionamiento ya explicado en el que el microcontrolador solo envía datos al detectar un patrón anómalo o, en su defecto, tras 60 segundos de inactividad para mantener la conexión viva mediante mensajes keepalive. Sin este filtrado, el sistema enviaría una media de más de 36.000 mensajes por sensor al día (10.2 Hz de muestreo durante 24 horas). Con la lógica de detección implementada, esta cifra se reduce a aproximadamente 203 mensajes por hora, lo que supone una reducción del tráfico superior al 94%.

4.3.2. OPTIMIZACIÓN DE LA RESPUESTA CON IA Y AHORRO ENERGÉTICO

La segunda fase de optimización se basa en predecir la respuesta del sistema con mayor precisión. En un sistema tradicional, cuando se detecta un evento anómalo (por ejemplo, una inclinación no esperada), la solución más simple es restablecer el actuador a su posición inicial, que suele corresponder a un valor PWM de 0. Sin embargo, esta aproximación no siempre representa la mejor solución física ni energética, ya que puede provocar correcciones innecesarias o excesivas.

Con el modelo de inteligencia artificial desarrollado, se calcula de forma más precisa qué valor PWM debe aplicarse en función del patrón de inclinación detectado. Esto permite adaptar la corrección a la magnitud y dirección del evento, evitando movimientos innecesarios.

Un ejemplo práctico: si el sistema detecta una inclinación de +15° en rot_x pero el resto de variables están estables, el modelo puede estimar que el valor óptimo de PWM para corregir esa desviación es 65. Si el sistema reaccionara sin optimización devolviendo siempre el actuador a 0 (posición original), se produciría un gasto energético innecesario y un ajuste excesivo, que podría incluso desestabilizar la estructura. La IA, en cambio, permite aplicar una respuesta intermedia y proporcional.

Este ajuste inteligente también repercute en la durabilidad del sistema, al reducir la cantidad y magnitud de los movimientos del actuador, se reduce el desgaste mecánico, se prolonga la vida útil del dispositivo y se mejora la eficiencia global del sistema. En un entorno donde los actuadores funcionan con energía solar y deben operar de forma autónoma, estas mejoras son críticas.

4.3.3. IMPACTO GLOBAL DE LA OPTIMIZACIÓN

La implementación conjunta de estas dos estrategias no solo mejora el rendimiento del sistema, sino que lo transforma en una red IoT inteligente: menos tráfico, decisiones más informadas, menor consumo energético, y mejor respuesta operativa. Todo el proceso puede representarse como un ciclo continuo de mejora, como se muestra en la Figura 8, donde los datos captados en tiempo real son analizados por el modelo de IA, se toman decisiones optimizadas, y estas se validan y retroalimentan continuamente al sistema. En los siguientes apartados se detallarán las herramientas empleadas para entrenar los modelos de IA, los resultados obtenidos en simulaciones, y cómo estas mejoras se traducen en mejoras cuantificables en la red.

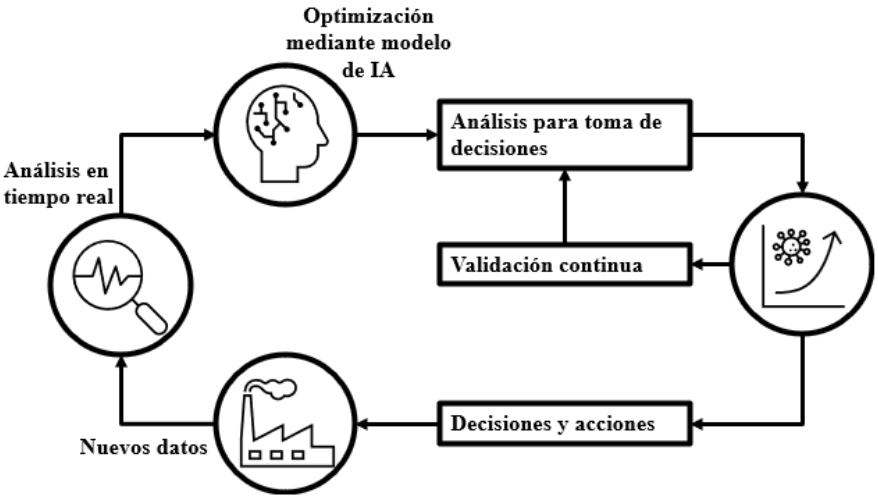


Figura 8: Ciclo de optimización del algoritmo

5. DISEÑO DE LA RED DE COMUNICACIONES

En este capítulo se describe el diseño completo simulado de la red de comunicaciones propuesta, desde la detección de eventos en el sensor hasta la transmisión, realizando el procesamiento mediante inteligencia artificial y enviando de la respuesta al actuador. El enfoque se ha estructurado para simular de forma modular el flujo de información en un sistema IoT, poniendo el foco en su optimización buscando así obtener una mayor eficiencia y respuesta más inteligente.

ESTUDIO PREVIO DE LOS DATOS DISPONIBLES

El diseño del sistema parte del análisis de un conjunto de datos reales obtenidos a partir de la simulación de un único sensor ADXL345 conectado a un microcontrolador ESP32. Aunque el objetivo del proyecto contempla una red de sensores a gran escala, esta fase inicial se ha centrado en un único nodo para estudiar en profundidad su comportamiento, sin perder de vista que se ha evaluado el rendimiento y la capacidad de red bajo condiciones de operación simultánea de todos los sensores, tal como se detalló en capítulos anteriores.

Es importante destacar que este dataset, de gran cantidad de datos (55.1 horas de simulación a 10.2 Hz, unas 2 millones de muestras), es representativo para explorar el tipo de comportamiento que se desea modelar, a pesar de tratarse de un solo sensor. Además, se diseñó con el objetivo explícito de capturar eventos de inclinación anómalos, por lo que presenta una densidad de eventos significativa. Esto resulta útil para entrenar y validar el sistema propuesto, ya que permite observar con claridad las reacciones del sistema ante perturbaciones, aunque puede introducir ciertos sesgos. No obstante, otro aspecto a tener en cuenta es el gran porcentaje de muestras sin evento, lo que también supone una limitación de cara al entrenamiento del modelo, aunque todo esto se detallará en el próximo capítulo.

Antes de abordar el diseño completo del flujo de comunicaciones en Simulink, es esencial comprender cómo se comportan los datos de entrada, qué tendencias siguen y en qué condiciones se producen los eventos. En la Figura 9 se muestra la evolución temporal de las principales variables:

- rot_x y rot_y: reflejan la inclinación del sensor en grados. Se observa que rot_x tiene una mayor variabilidad e incidencia en la detección de eventos.
- grav: representa el módulo de la aceleración total. Permanece relativamente estable, aunque muestra picos que pueden estar asociados a vibraciones o impactos puntuales.
- PWMValue1: es la respuesta aplicada por el sistema. Se evidencia que solo en ciertos picos de rot_x se genera una respuesta significativa (valores elevados de PWM), mientras que la media de esta variable permanece cercana a cero.

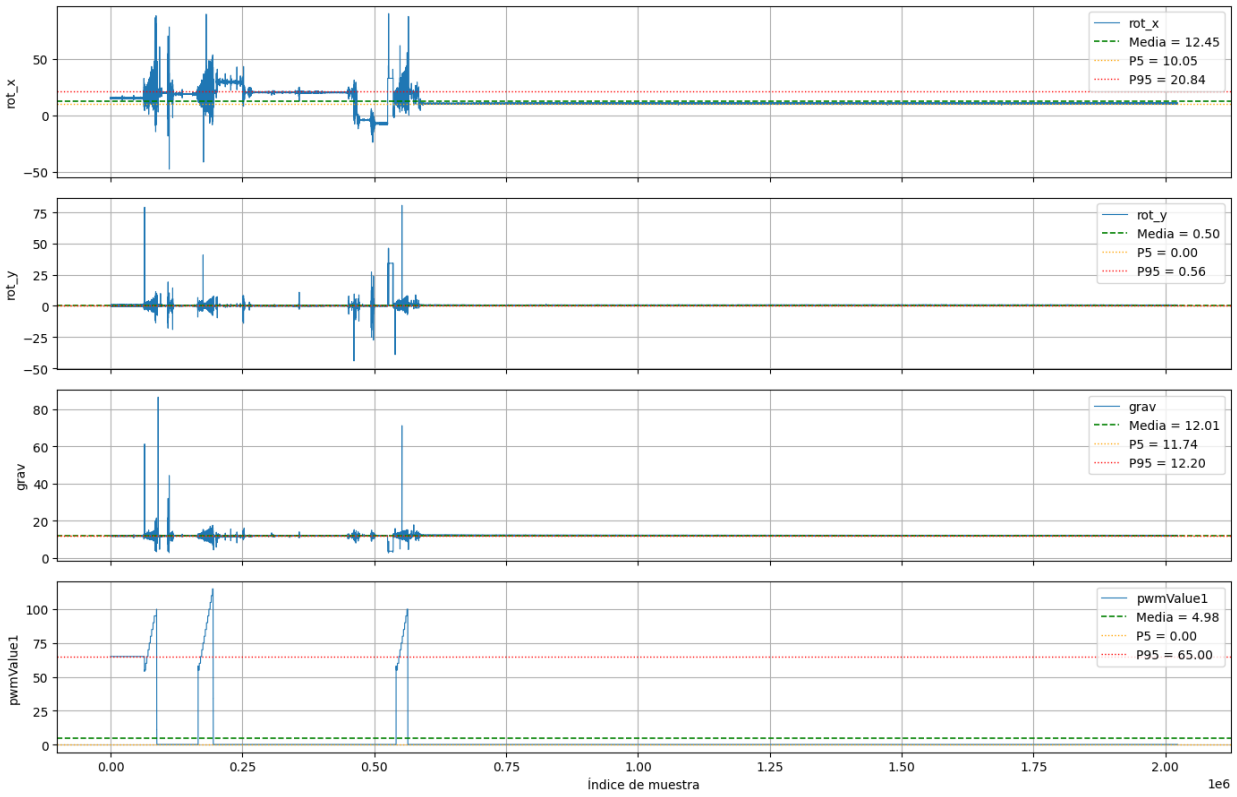


Figura 9: Evolución temporal de variables físicas

A la vista de los datos, puede apreciarse que la gran mayoría del tiempo no se producen incidencias en el sensor. Las variables rot_x, rot_y y grav se mantienen estables en torno a sus valores medios, y el valor de PWMValue1 (valor real del PWM) permanece cercano a cero, indicando que no se aplican correcciones. Este comportamiento refuerza la necesidad de aplicar estrategias inteligentes para filtrar transmisiones innecesarias, ya que la mayor parte del tiempo el sistema permanece en estado normal.

Solo en instantes puntuales, visibles como picos en las gráficas, se detectan eventos que disparan el envío de datos y generan una respuesta en el sistema (aplicación de un PWM significativamente distinto de cero). Analizar de forma cuantitativa este patrón nos permitirá definir umbrales y establecer la lógica de funcionamiento del sistema. Los eventos, cuando ocurren, tienen una duración breve pero generan una respuesta del actuador. El valor PWM aplicado se mantiene en rangos altos durante la corrección, mientras que fuera de evento permanece prácticamente en cero.

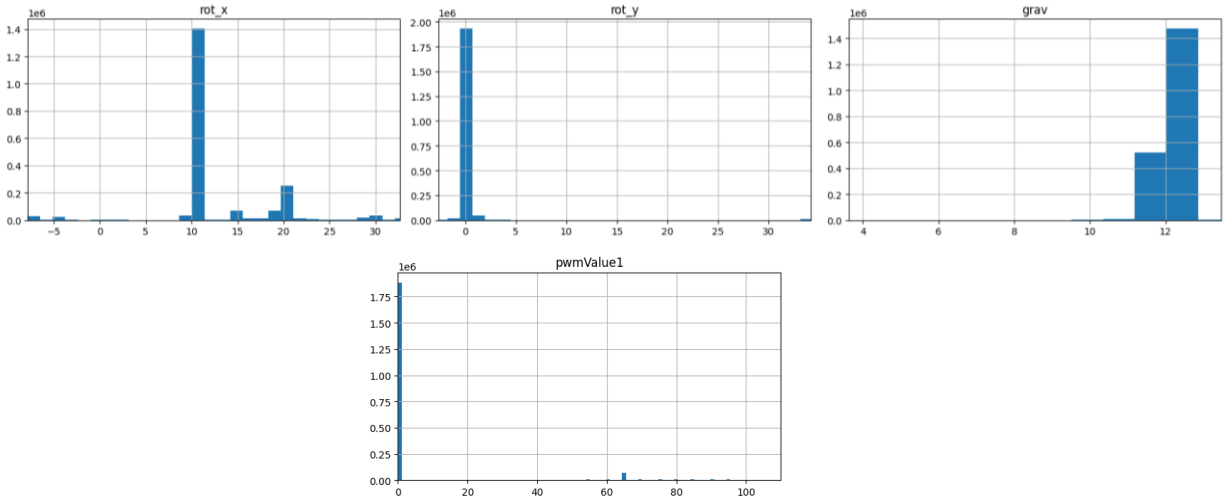


Figura 10: Histogramas de frecuencias de las variables representativas

La Figura 10 presenta los histogramas de frecuencia para las cuatro variables representativas más relevantes en el sistema: rot_x, rot_y, grav y PWMValue1. Estos histogramas permiten observar de forma clara la distribución general de los valores registrados durante la simulación, proporcionando una base sólida para entender el comportamiento del sistema y guiar el diseño del modelo de inteligencia artificial.

La variable rot_x, que representa la inclinación del sensor en el eje x (principal eje de inclinación del panel), muestra una fuerte concentración alrededor de los 12 grados. Este valor es coherente con el diseño del experimento, ya que gran parte del tiempo el sistema se encuentra en un estado estable, con ligeras oscilaciones. Sin embargo, también se observan repuntes más altos (alrededor de 20 y 30 grados), que corresponden a eventos anómalos simulados. Esta doble distribución indica claramente que es una variable clave para detectar desviaciones o situaciones de riesgo.

En el caso de la variable rot_y, los valores se agrupan fuertemente en torno al 0, con una distribución mucho más estrecha. Esto es consistente con lo observado previamente: el sistema presenta muy poca inclinación en el eje y, lo cual tiene sentido si se considera que las principales rotaciones y fuerzas simuladas afectan al eje x, y no al eje lateral. Rot_y puede considerarse por tanto como una variable con un papel más secundario en la detección de eventos.

El módulo de la aceleración (grav) se concentra principalmente entre 11.5 y 12.5 m/s², lo cual en línea con el valor esperado de la aceleración gravitacional (9.8 m/s²) ajustado a la calibración del sensor, ya que aunque grav esté centrado en torno a 12 m/s², esto no representa un comportamiento anómalo, sino que refleja la calibración aplicada al sensor y su orientación física. Esto refuerza la idea de que el sensor se encuentra correctamente orientado y que la mayor parte del tiempo está registrando una situación normal. Las desviaciones son escasas, lo que confirma que los eventos anómalos son excepcionales y no representan el comportamiento estándar del sistema.

Finalmente, la variable que representa el valor PWM de consigna enviado al actuador en la simulación realizada muestra una fuerte concentración en 0, lo cual es coherente con el hecho de que solo se genera un valor distinto de cero en caso de evento anómalo. Sin embargo, también se observan valores en torno a 60-80, y muy pocos valores en otros puntos, lo que sugiere que en ciertas ocasiones el sistema ha detectado eventos que requieren corrección y ha generado una respuesta proporcional y en un rango medio, probablemente más por defecto que por un procedimiento razonado.

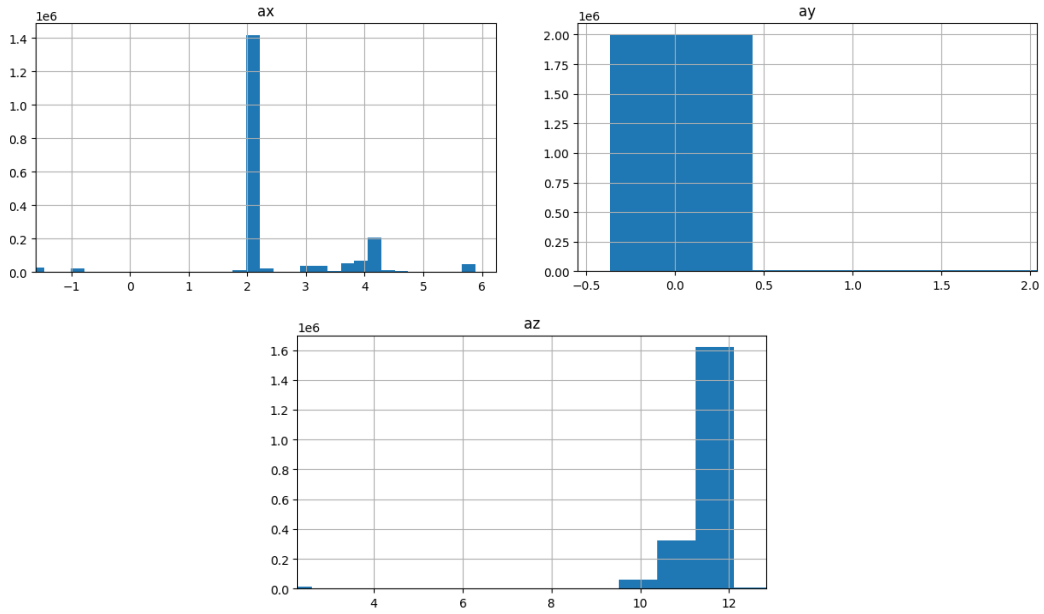


Figura 11: Histogramas de frecuencias de las variables reconstruidas

Las distribución observadas en los histogramas de las variables reconstruidas, los cuales se pueden observar en la Figura 11, podemos llegar a la conclusión de que los valores de ax, ay y az tienen coherencia desde el punto de vista físico, especialmente si se considera cómo se sitúa el sensor ADXL345 en una estructura estática como un panel solar. Estos valores se han obtenido a partir de las fórmulas especificadas anteriormente para el cálculo de las rotaciones en los ejes y la gravedad (detalladas en la página 24).

- az: Es la componente que suele alinearse con el eje perpendicular a la superficie del panel (vertical respecto al sensor). En condiciones normales, cuando el panel está correctamente nivelado o levemente inclinado, la fuerza gravitacional de 1g (≈ 9.8 m/s²) actúa mayoritariamente sobre este eje. Por tanto, es lógico que la mayoría de los valores de az se concentren entre 10 y 12, tal como muestra el histograma. Esto refleja un estado de equilibrio estructural.

- ax: Esta componente representa la proyección de la gravedad sobre el eje longitudinal del sensor. Dado que los paneles solares apenas rotan en esta dirección (salvo pequeños ajustes), es razonable que ax se mantenga en valores bajos o con pequeñas

desviaciones positivas, como las observadas (centro en torno a 2 m/s²). Las pequeñas variaciones reflejan inclinaciones detectadas durante los eventos simulados.

- ay: En muchos casos, este eje representa el eje transversal, perpendicular al plano del panel. La práctica totalidad de los datos aparecen cercanos a cero, lo cual es coherente si el panel apenas presenta inclinaciones significativas en este eje (es decir, no hay giros laterales ni torsiones detectadas en ese eje durante la simulación). Esto concuerda con una instalación estable, donde la mayoría de las variaciones estructurales o por viento se producen en el eje de inclinación principal (rot_x) y no lateral.

El hecho de que ax y az concentren la mayoría de las magnitudes y que ay se mantenga prácticamente constante confirma que:

- La gravedad se proyecta principalmente en el eje Z del sensor, como es esperable en una superficie horizontal.
- Las variaciones de inclinación captadas por el sensor corresponden mayoritariamente a rot_x, ya que implican variaciones relevantes en ax, pero no en ay.
- La estructura permanece estable en su eje transversal (poca actividad en rot_y y ay), lo cual valida el comportamiento físico simulado.

Este análisis es muy útil porque reafirma que la orientación del sensor y su respuesta frente a inclinaciones se han interpretado correctamente, lo que será fundamental al entrenar modelos de inteligencia artificial en los siguientes apartados. También refuerza que rot_x y ax son los principales indicadores de eventos relevantes, mientras que rot_y y ay presentan un comportamiento basal mucho más estable.

Estos datos e histogramas respaldan las decisiones tomadas en el diseño del sistema, tanto en cuanto a la lógica de detección de eventos como en la respuesta del modelo de inteligencia artificial. El sistema está calibrado para estar inactivo la mayor parte del tiempo (modo ahorro), y solo responder ante desviaciones detectadas por rot_x y validadas mediante los patrones de grav. Esta información resulta crucial antes de abordar la modelización IA y el diseño completo de la red de comunicaciones en Simulink.

SIMULACIÓN DEL ENVÍO DE DATOS Y DE LORAWAN

La simulación parte directamente de los datos recogidos en la simulación real del sensor, que se importan al entorno de Simulink desde el workspace de Matlab y a través de la variable simin_ts. Esta variable contiene las variables rot_x, rot_y y grav formuladas como series temporales.

Antes de la modulación CPM, que actúa como sustituto funcional de la modulación LoRaWAN para simplificar el sistema y cuya elección se justifica en este apartado más adelanre, el flujo de Simulink busca reproducir el comportamiento lógico que ejecutaría un microcontrolador real en el sistema físico. Debido a las exigencias del entorno Simulink, se incluyen varios bloques intermedios, que serán explicados a continuación, para garantizar la compatibilidad entre dimensiones, tipo de dato y estructura. En la Figura 12 podemos observar en detalle este primer tramo del flujo simulado.

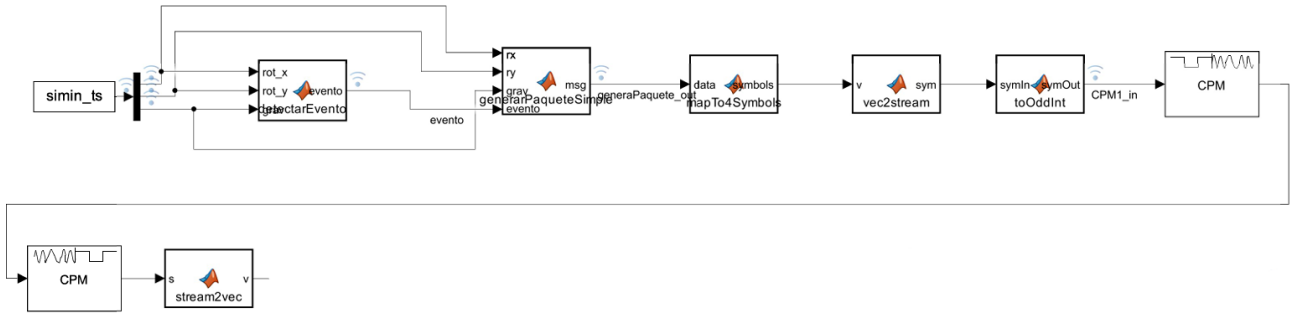


Figura 12: Flujo de Simulink del envío de datos y la modulación CPM

Tras cargar los datos necesarios en Simulink, el primer bloque será el encargado de manejar la lógica de detección, tal como ocurriría en el microcontrolador ESP32. Esta se implementa mediante la función detectarEvento, que evalúa tres condiciones sobre las variables físicas calculadas a partir de los datos del sensor: rotación en X (rot_x), rotación en Y (rot_y) y módulo de la aceleración (grav). Estas condiciones reflejan umbrales físicos que representan situaciones de inclinación anómala o posibles movimientos bruscos del sensor:

```
function evento = detectarEvento(rot_x, rot_y, grav)
  evento = 0;
  if rot_x > 25 || rot_x < -5
     evento = 1;
  elseif abs(rot_y) > 2.7
     evento = 1;
  elseif grav < 11.0 || grav > 12.6
     evento = 1;
  end
end
```

Figura 13: Código de la función para detectar evento en Simulink

El diseño de la función detectarEvento parte del análisis estadístico de las variables registradas. Las condiciones definidas se basan en la distribución de los datos:

- rot_x > 25 o < -5: umbrales situados fuera del percentil 95 y 5 respectivamente, lo que implica inclinaciones muy poco frecuentes.

- abs(rot_y) > 2.7: valor significativamente superior al percentil 95 (P95 = 0.56), lo que indica un movimiento anómalo en ese eje.

- grav < 11.0 o > 12.6: también fuera del rango P5-P95 (11.74 – 12.20), útil para identificar caídas, impactos o pérdidas de estabilidad.

Esto permite establecer reglas sólidas para detectar eventos con base en comportamientos infrecuentes. La Tabla 3, a continuación, resume los valores estadísticos clave obtenidos mediante un análisis estadístico realizado en Python a partir del dataset original.

Variable	Media	P5	P50	P95	Máx	Mín
rot_x	12.45	10.05	10.30	20.84	90.00	-47.85
rot_y	0.50	0.00	0.37	0.56	80.60	-44.21
grav	12.01	11.74	12.11	12.20	86.62	2.81
PWMValue1	4.98	0.00	0.00	65.00	115.00	0.00
ax	2.55	2.12	2.16	4.16	11.18	-11.81
ay	0.08	0.00	0.08	0.12	70.29	-10.00
az	11.67	11.02	11.92	11.99	86.51	0.00

Tabla 3: Principales valores estadísticos de las variables destacadas

Estos datos muestran que las condiciones definidas representan efectivamente valores extremos, justificando su uso como disparadores de transmisión en una red de sensores optimizada para bajo consumo y eficiencia de red.

A continuación, el bloque generaPaqueteSimple lleva a cabo la lógica de empaquetado de los datos. Se genera un paquete de la forma [flag, rot_x, rot_y, grav], donde flag = 1 indica evento y flag = 0 indica un mensaje de tipo keepalive. Si no se detecta ningún evento y han transcurrido 60 segundos desde la última transmisión, se genera un mensaje keepalive para confirmar que el nodo sigue activo. Esto simula también la lógica que seguiría el microcontrolador.

Una vez generado el paquete, este se convierte a símbolos utilizando el bloque mapTo4Symbols, que cuantiza los valores para adaptarlos al canal de transmisión. A continuación, vec2stream transforma la secuencia de símbolos en una señal serializada. Esta secuencia pasa por toOddInt, que adapta los valores a la entrada requerida por el bloque de modulación CPM. Tras la creación del paquete y su correspondiente procesamiento, ahora pasaríamos a la lógica que simula la comunicación LoRaWan, Esta consta de dos fases: modulación y demodulación. Ambas serán explicadas a continuación a pesar de que en medio se realizaría la simulación de MQTT y del procesamiento de los datos con el algoritmo de Inteligencia Artificial.

5.2.1. ENVÍO DE LOS DATOS AL BLOQUE DE IA

El bloque CPM Modulator Baseband se encarga de realizar la modulación de los datos previamente convertidos a símbolos, emulando así el proceso de transmisión de información por vía inalámbrica. En este trabajo se ha optado por utilizar modulación CPM (Continuous Phase Modulation) como una aproximación simplificada a la modulación CSS (Chirp Spread Spectrum) empleada por la tecnología LoRa. Esta elección se debe, en primer lugar, a la ausencia de bloques propios de Simulink que reproduzcan directamente el comportamiento de LoRa, lo que haría necesaria una implementación matemática compleja fuera del alcance del presente trabajo. En segundo lugar, CPM permite simular de manera adecuada entornos ruidosos y mantener ciertas características funcionales como la robustez frente a interferencias y la eficiencia espectral, propiedades clave también en sistemas reales basados en LoRaWAN [16].

Los parámetros seleccionados para el bloque CPM han sido los siguientes:

- M-ary number = 4: Se utiliza una modulación en 4 niveles (cuaternaria), lo cual permite representar eficientemente los datos del sistema en forma de símbolos discretos, sin necesidad de ampliar excesivamente el número de intervalos de transmisión ni la complejidad de los bloques de modulación y demodulación. Es importante tener en cuenta que internamente las variables físicas como rot_x, rot_y y grav son del tipo double. Para transmitirlas a través del canal simulado, es necesario cuantizarlas y discretizarlas, convirtiéndolas en símbolos representables por la modulación CPM, por eso es necesario el procesamiento de los datos que hemos explicado antes.

- Pulse length = 1 símbolo y Symbol prehistory = 1: Estos parámetros indican que la fase de la señal se actualiza en cada símbolo, sin memoria adicional, lo cual reduce la latencia y facilita la la demodulación, especialmente para paquetes cortos y eventos puntuales, como los que generan los sensores.

- Output type = double: Se ha seleccionado este tipo de dato para representar con mayor precisión la señal modulada. Al trabajar en un entorno simulado, la prioridad también está en la correcta interpretación y propagación de la señal a través del canal.

En conjunto, esta configuración permite implementar una modulación sencilla y eficazen Simulink, de manera coherente y fiel a la red planteada. Aunque no pretende replicar exactamente el comportamiento espectral de LoRa, sí ofrece una representación que permite estudiar el impacto de los eventos detectados, la transmisión de paquetes y la respuesta del sistema de inteligencia artificial en la nube en un sistema lo más realista posible.

Tras la modulación CPM, se realiza la etapa de demodulación utilizando el bloque correspondiente en Simulink, que permite recuperar la secuencia original de símbolos transmitidos. Esta secuencia es reconstruida después mediante el bloque stream2vec, encargado de reagrupar los símbolos en paquetes estructurados, listos para ser procesados

por los bloques que simulan el funcionamiento del sistema en la nube. Aunque en una implementación más realista se incluiría un canal con ruido (como el canal AWGN) para modelar las posibles interferencias en la transmisión, en esta simulación se ha decidido omitirlo. Esta decisión responde a la necesidad de mantener el sistema lo más limpio y estable posible, ya que el foco principal de este trabajo no reside en el estudio detallado del canal físico de comunicación, sino en la optimización de la red IoT a través de técnicas de inteligencia artificial. Por tanto, se ha priorizado una simulación funcional que represente fielmente el flujo de datos, sin añadir distorsiones que dificulten la validación de los modelos desarrollados en las etapas posteriores. En el siguiente apartado se describe cómo se continúa este flujo con la simulación de la arquitectura MQTT y el procesamiento de los datos mediante algoritmos inteligentes en la nube.

5.2.2. ENVÍO DE LA RESPUESTA AL ACTUADOR

Una vez que los datos provenientes del sensor han sido procesados por el modelo de inteligencia artificial en la nube, el sistema genera una respuesta correctiva. Esta respuesta, generalmente un valor PWM (modulación por ancho de pulso), representa la nueva posición que debe adoptar el actuador físico asociado al sensor, como puede ser un servo encargado de ajustar la inclinación de una estructura.

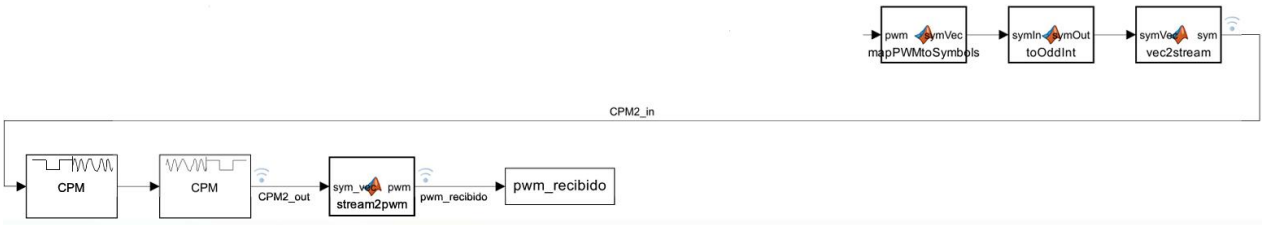


Figura 14: Simulación de la demodulación de CPM y la recepción final de los datos

Tal como se puede ver en la Figura 14, el valor de salida es tratado como un nuevo paquete de datos que debe recorrer el camino inverso: desde el entorno de procesamiento en la nube hasta el microcontrolador físico. Para simular este proceso, se sigue un esquema idéntico al de la transmisión ascendente, utilizando nuevamente modulación CPM para emular el envío de la información.

Primero, el valor PWM generado por la IA es transformado a un formato de símbolo compatible con el modulador CPM. Una vez modulado, el paquete simulado atraviesa el canal (en este caso, sin añadir ruido), y se procesa mediante un bloque de demodulación CPM Demodulator Baseband, que permite recuperar los símbolos originales transmitidos. Esta etapa es crucial para evaluar que el sistema puede recibir respuestas precisas incluso tras la conversión y transmisión simbólica.

Posteriormente, el bloque stream2vec reconstruye el vector original a partir de los símbolos demodulados. Este vector es interpretado como la consigna final para el actuador

correspondiente. En una implementación física real, este valor se transmitiría desde el microcontrolador al actuador mediante una señal PWM.

En el diseño planteado, esta señal PWM se simula como un ángulo calculado a partir de una señal de entrada analógica. En concreto, el microcontrolador realiza una lectura mediante la función analogRead, que devuelve un valor entre 0 y 1023. Esta lectura se transforma a un rango de 0 a 180 grados usando la función map(value, 0, 1023, 0, 180), que representa la posición deseada del actuador. El servo físico traduce este valor a una posición mecánica con un rango de PWM aproximado de 0 a 120. Este rango es suficiente para cubrir el espectro de inclinaciones comunes en estructuras fotovoltaicas u otros sistemas sensibles a orientación [23].

Cabe destacar que, aunque en esta simulación el actuador no ha sido modelado en detalle, el sistema propuesto sí contempla su interacción lógica con el microcontrolador. Este diseño modular permite extender fácilmente la simulación para incluir modelos más complejos de respuesta física, sin alterar la arquitectura general.

En resumen, esta parte de la simulación valida que el sistema puede cerrar correctamente el ciclo de comunicación: desde la detección del evento por el sensor hasta la ejecución de una acción correctiva. La transmisión de este valor PWM a través de bloques de modulación y demodulación representa una abstracción funcional del comportamiento que se esperaría en una red IoT real, facilitando el análisis de precisión, latencia y eficiencia del sistema propuesto.

SIMULACIÓN DE MQTT Y EL PROCESAMIENTO DE DATOS CON IA

Teniendo en cuenta que el objetivo principal del proyecto es evaluar la efectividad del sistema de inteligencia artificial en la optimización del comportamiento de una red IoT, poniendo en este análisis el foco y dándole prioridad, se ha optado por dividir la simulación completa en dos bloques independientes:

El primer bloque, representado en la Figura 15, corresponde a la simulación del proceso a partir de la transmisión de lo que sería MQTT hasta la nube y de regreso al actuador. Esta simulación tiene como objetivo verificar que el flujo de información sea funcional y que todos los componentes del sistema, sensor, microcontrolador, gateway, servidor, procesamiento en la nube y actuador, stén correctamente sincronizados a nivel lógico. Aunque no se evalúa aquí la toma de decisiones inteligente, sí se valida que el diseño sea capaz de enviar, recibir y reenviar datos de forma coherente y robusta.

Para representar este proceso en Simulink, y ante la imposibilidad de integrar directamente una infraestructura MQTT real, se ha optado por utilizar bloques de lectura y escritura en memoria (Write to Memory y Read from Memory). Estos bloques tienen como objetivo simular el comportamiento de un broker MQTT, que en un sistema real actuaría como

intermediario entre los distintos componentes de la red, permitiendo una comunicación asíncrona. En la simulación, la información que sale de la parte del sistema correspondiente al servidor LoRaWAN (modulación CPM) y que se desea enviar al bloque que simula el servidor de inteligencia artificial se escribe en una zona de memoria compartida. Posteriormente, el servidor de IA (simulado mediante un bloque Matlab Function) accede a esta misma memoria para leer los datos.

Este esquema permite simular de forma eficaz el comportamiento publish-subscribe típico de MQTT, manteniendo la lógica de comunicación real, pero evitando complicaciones excesivas que se salen del margen contemplado dentro de este trabajo.

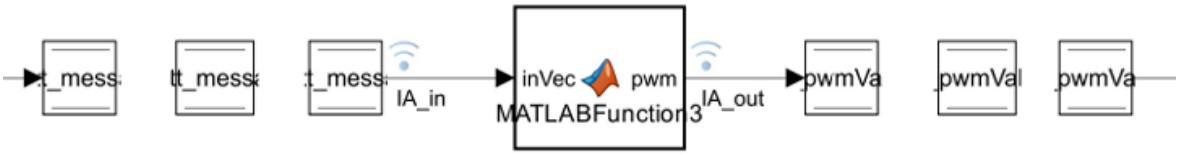


Figura 15: Simulación de MQTT y el procesamiento de datos con IA

El segundo bloque, mostrado en la Figura 16, se trata de una simulación independiente y centrada exclusivamente en el procesamiento con IA, donde se analiza el comportamiento del algoritmo ante eventos detectados, sin incluir la complejidad de la transmisión.

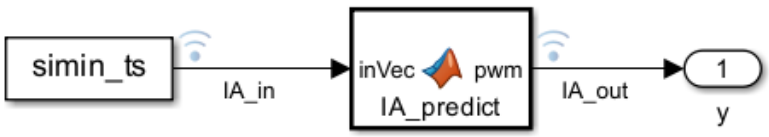


Figura 16: Simulación final del procesamiento con IA

Esta decisión responde a varias limitaciones prácticas observadas durante el desarrollo. Al intentar integrar el procesamiento completo en una sola simulación, surgieron problemas relacionados con la gestión de tipos de datos, errores en la conversión de estructuras y, sobre todo, tiempos de cómputo extremadamente elevados. Esto hacía inviable validar todo el sistema en una única ejecución, ya que una llamada reiterada a la función de predicción en cada muestra suponía una carga computacional excesiva.

En consecuencia, tal como se ha explicado, el bloque de inteligencia artificial se ha implementado de forma separada, permitiendo así controlar mejor el entorno de prueba y observar el efecto directo del modelo sobre las variables físicas relevantes (rot_x, rot_y y grav). Las gráficas que se muestran a continuación ilustran el comportamiento del PWM simulado tras el procesamiento realizado por la IA frente al PWM real, centrado en un tramo del conjunto de datos en el que se registran eventos. Este tramo ha sido seleccionado intencionadamente como muestra representativa, debido a que en Simulink no es posible representar el comportamiento completo del sistema con más de 2 millones de muestras en un tiempo razonable, y solo es viable trabajar por segmentos.

El código correspondiente a la función IA_predict, que recibe la variable simin_ts y devuelve el valor de PWM predicho se encuentra en el Anexo II, así como la función de Python a la que llama y donde se encuentra el modelo programado.

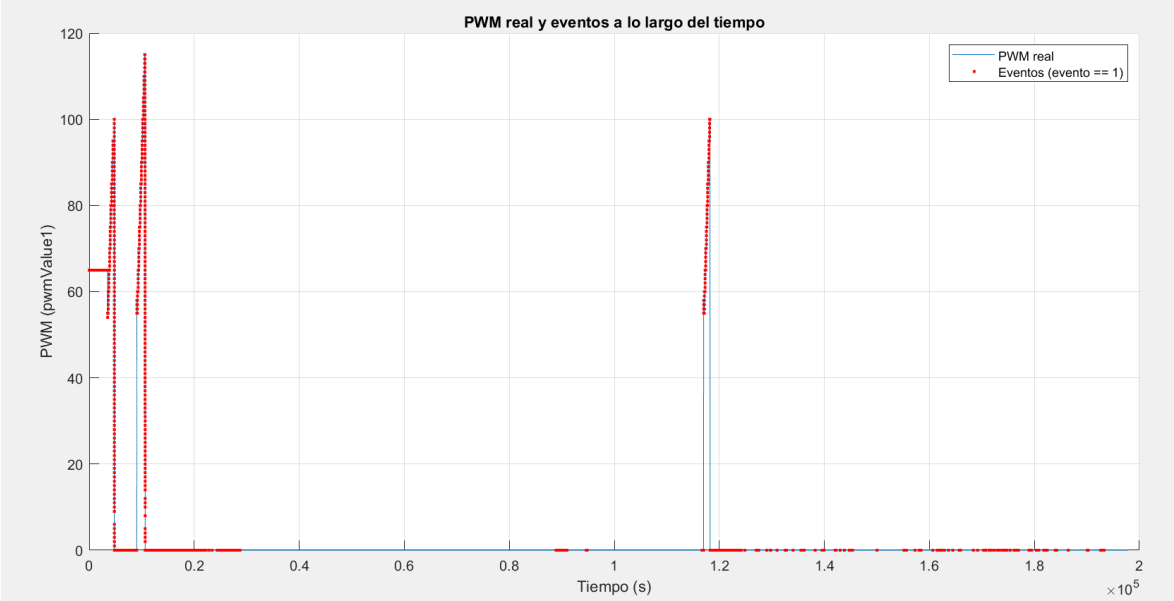


Figura 17: Valores de PWM real con eventos señalados

Como se puede observar en la Figura 17, existe una fuerte correlación entre la activación de eventos (evento = 1) y las variaciones del valor PWM. En condiciones normales (evento = 0), el PWM se mantiene prácticamente siempre en cero, mientras que en los tramos con eventos se registran valores elevados y transiciones bruscas.

En la Figura 18 podemos ver una franja concreta de tiempo en los datos en la que podemos observar presencia de eventos. El valor PWM generado por el modelo de inteligencia artificial presenta una respuesta con muchos más pico y de forma continua que el valor real, el cual sigue una evolución escalonada típica del sistema físico original. Es importante tener en cuenta que el modelo ha sido diseñado para emitir una única señal de respuesta por evento detectado, no para adaptarse dinámicamente a una secuencia continua de activaciones. En este sentido, la persistencia del flag de evento durante varios segundos no estaba prevista como una situación a replicar de forma secuencial. Por tanto, aunque en apariencia el modelo no refleje la evolución paso a paso del PWM real, sí identifica correctamente la fase de activación y mantiene el valor dentro de un rango válido y funcional.

Esta diferencia en la evolución no representa un problema grave: lo relevante en este contexto es que la IA detecte cuándo actuar y proponga una respuesta realista, no que emule al detalle cada paso intermedio. Dado que un error de ±5 unidades en PWM apenas tiene impacto apreciable en el funcionamiento de un actuador como un servo, la precisión obtenida resulta más que suficiente.

Sin embargo, de cara a una futura implementación física, más allá del enfoque teórico y de la simulación que se han utilizado en este trabajo, sería necesario incorporar mecanismos de control adicionales que permitan gestionar eventos sostenidos o fluctuaciones persistentes en los datos. Por ejemplo, se podría condicionar la frecuencia de activación del modelo de IA o introducir una lógica de suavizado que evite múltiples llamadas innecesarias al sistema de decisión durante un único evento prolongado. Este tipo de control sería especialmente relevante para garantizar una respuesta coherente y eficiente en entornos reales, donde el actuador opera de forma continua y se busca maximizar tanto la estabilidad como el ahorro energético del sistema. Aunque en este trabajo no se ha priorizado dicha implementación, se reconoce como una mejora razonable y alineada con una futura aplicación práctica.

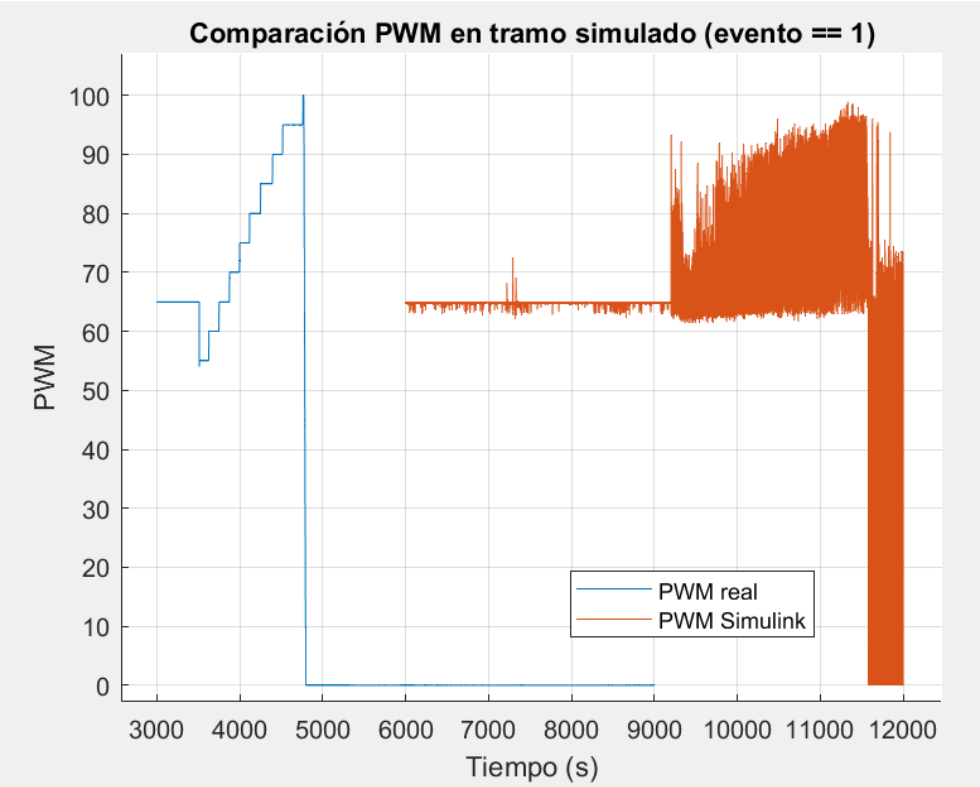


Figura 18: Comparación resultado PWM (retrasado) con el real

A continuación, se presentan algunas métricas extraídas del análisis de datos en las que se compara el PWM obtenido de la simulación de Simulink vs los valores reales. La comparativa entre el PWM generado por Simulink y el real solo tiene sentido durante los eventos (evento = 1), ya que es en esos momentos cuando el sistema debe actuar y la inteligencia artificial entra en funcionamiento. A continuación, podemos analizar algunas métricas relevantes. Cabe destacar que este fragmento no se ha escogido por su buen ajuste a los valores reales, sino por tener uno de los eventos más destacables, lo cual se ha considerado interesante para analizar en este trabajo. Sin embargo, en el siguiente capítulo se estudiarán de manera más exhaustiva todos los distintos tramos de los datos obtenidos.

- PWM Simulink: $\mu = 65.16$ $\sigma = 1.79$
  El modelo predice valores de PWM estables, cercanos a un valor medio elevado y con muy poca variabilidad.

- PWM Real: $\mu = 60.94$ $\sigma = 30.26$
  Los valores reales muestran mayor dispersión, con un rango más amplio de respuesta y picos más irregulares, lo que se justifica con la explicación previa en lo referente a valores continuos.

- MAE (Mean Absolute Error) = 20.52
  En promedio, la diferencia absoluta entre la predicción del modelo y el valor real es de 20.52 unidades PWM.

- RMSE (Root Mean Squared Error) = 31.05
  El error cuadrático medio, que penaliza más los errores grandes, es de 31.05, reflejando una cierta distancia entre la respuesta real y la estimada.

Métrica	**Valor general**	**Valor (evento = 0)**	**Valor (evento = 1)**
Muestras totales	2.023.188	1.727.345	295.843
Media	4.98	0.00	34.03
Desviación	18.72	0.07	37.52
Mínimo	0.00	0.00	0.00
Máximo	115.00	33.00	115.00
Percentil 5	0.00	0.00	0.00
Percentil 95	65.00	0.00	65.00

Tabla 4: Estadísticas del PWM real (PWMValue1)

Estos resultados demuestran que el modelo IA es capaz de capturar patrones relevantes del comportamiento del PWM, aunque el rango completo de variación aún no se replica con total precisión. Aun así, es importante subrayar que estas simulaciones representan únicamente una parte del flujo completo, y los resultados más completos se analizarán más adelante una vez descrito con detalle el desarrollo del algoritmo de inteligencia artificial.

Sin embargo, de estos resultados se pueden ir extrayendo ciertas conclusiones clave que serán relevantes para el posterior análisis:

- Alta precisión media del modelo IA: La media del PWM predicho en Simulink (65.16) está razonablemente alineada con el valor medio del PWM real durante los eventos (60.94), lo cual indica que el modelo de inteligencia artificial es capaz de capturar el patrón base de respuesta del sistema. Aunque existe una diferencia de

aproximadamente 4.2 unidades, esta se encuentra dentro de un margen tolerable en aplicaciones industriales donde se permiten ciertas tolerancias operativas.

- Dispersión significativamente menor en la predicción: La desviación estándar del PWM predicho por la IA ($\sigma = 1.79$) es mucho menor que la del valor real ($\sigma = 30.26$), lo que indica que el modelo genera respuestas más estables y menos ruidosas. Este comportamiento tiene una doble interpretación. Por un lado, supone una ventaja desde la perspectiva de la seguridad operativa, ya que reduce las oscilaciones bruscas que podrían dañar el actuador o comprometer la estructura. Por otro lado, esta baja dispersión también se debe a que el sistema real opera con valores escalonados dentro del margen de movimiento permitido por un actuador físico. Aunque en el modelo diseñado no se ha buscado imitar ese comportamiento mecánico, es positivo comprobar que el modelo predice valores en un rango ajustado durante un evento. En apartados posteriores se mostrará cómo, en simulaciones completas, el modelo es capaz de adaptarse dinámicamente dentro de ese rango, manteniendo coherencia con el sistema real.

  Además, en sistemas físicos reales, es habitual que, ante la duda, por ejemplo, al finalizar un evento o en presencia de lecturas ambiguas, el controlador envíe el actuador a una posición segura, como PWM = 0 [14]. Esta estrategia busca minimizar riesgos, pero también puede generar movimientos innecesarios o poco eficientes. En cambio, la IA tiende a mantener una respuesta más progresiva, sin transiciones tan abruptas a cero, lo que favorece un comportamiento más continuo. En apartados posteriores se mostrará cómo, en simulaciones completas, el modelo es capaz de adaptarse dinámicamente dentro de ese rango físico, manteniendo la coherencia con el sistema real y aportando mayor estabilidad a la operación.

- Error absoluto y cuadrático a considerar: Las métricas MAE (20.52) y RMSE (31.05) son elevadas en comparación con la escala de valores de PWM (máximo 115), lo que evidencia que aún existe margen de mejora. Esta diferencia se explica en parte porque los eventos reales muestran un comportamiento abrupto (como se ha visto en la Figura 16), con PWM que pasan rápidamente de 0 a 100 y luego caen a 0. El modelo IA en Simulink, por el contrario, responde de forma más gradual y suavizada. También es importante destacar que esto corresponde a un fragmento concreto de los datos y que en un análisis más genérico se obtienen mejores resultados. Por otro lado, aunque el error es elevado, tampoco supondría dramático de cara a la aplicación física ya que el movimiento de un servo no se vería demasiado afectado por la diferencia entre valores.

- Importancia del filtrado previo: Dado que en el 85% de las muestras no hay evento y el PWM es 0 en prácticamente todos esos casos (media 0.00 y percentil 95 igual a 0), el modelo debe intentar evitar sesgos derivados de este desequilibrio en el dataset. Esto resalta la necesidad de separar los eventos y aplicar técnicas de balanceo o centrar el aprendizaje durante el entrenamiento del modelo, como se discutirá en el capítulo de diseño del algoritmo.

- Representatividad del tramo simulado: Aunque en Simulink se ha trabajado con un único tramo por limitaciones de carga computacional, el segmento ha sido cuidadosamente seleccionado para contener eventos con patrones reales de activación. Los resultados obtenidos, por tanto, son válidos como prueba de concepto, aunque, como se ha mencionado, se va a extender el estudio más tramos y a condiciones variadas.

En conjunto, estos resultados respaldan que el modelo IA implementado tiene un comportamiento prometedor para replicar decisiones reales en tiempo de evento. La simplificación del flujo en Simulink ha permitido centrarse en validar el funcionamiento general del sistema y detectar sus límites actuales. Los detalles sobre el algoritmo de IA, su estructura, entrenamiento y mejora de rendimiento se desarrollarán en el siguiente capítulo, donde se analizará con mayor profundidad cómo se pueden ajustar las predicciones para cubrir mejor la variabilidad real del PWM y lograr una optimización más precisa de la red de sensores.

RESUMEN DE LA SIMULACIÓN DEL SISTEMA

La Tabla 5 resume de forma esquemática el flujo completo de la simulación implementada en Simulink, detallando el rol de cada bloque, el tipo de procesamiento que realiza y el formato de datos que maneja. Este diseño refleja el comportamiento del sistema IoT propuesto, desde el origen de los datos hasta la transmisión de la respuesta correspondiente al actuador, incluyendo todas las etapas de codificación, transmisión, procesamiento y modulación involucradas en el proceso, pudiendo así visualizarlo de manera más sencilla.

Bloque	Función principal	Formato de datos que maneja
SIMIN_TS	Importa los datos ya procesados por el microcontrolador; simula su comportamiento.	rot_x, rot_y, grav (double)
DETECTAR EVENTO	Aplica umbrales físicos y filtra con una mediana. Si se detecta un movimiento anómalo, genera flag = 1.	flag (0 o 1, entero)
GENERAR PAQUETE	Construye el paquete [flag, rot_x, rot_y, grav].	Vector 1×4 (double)
TO ODD INT	Convierte cada componente del paquete a un entero impar.	Vector (enteros)

MAP → 4 SYMBOLS	Mapea los enteros a 4 símbolos (modulación cuaternaria tipo QPSK).	sym[4]
VEC → STREAM	Serializa los símbolos: genera un flujo de transmisión de símbolos.	Flujo: 200 símbolos/s
CPM Tx	Modula los símbolos usando CPM (Continuous Phase Modulation).	Onda modulada (banda base RF)
CPM Rx	Demodula la señal CPM recibida.	Símbolos discretos
STREAM → VEC	Reagrupa los símbolos demodulados en el paquete original.	Vector 1×4
IA (Python)	Si flag == 1, se llama a la función predict_PWM que genera un PWM entre 0 y 120.	PWMValue (entero)
MAP PWM → SYM	Codifica el valor PWM como un símbolo único.	sym[1]
TO ODD INT (ret)	Convierte el símbolo del PWM en un entero impar.	Entero
VEC → STREAM (ret)	Serializa el símbolo de retorno.	Flujo de un único símbolo
CPM Tx (ret)	Modula la respuesta con CPM para enviarla al actuador.	Onda modulada (RF)
CPM Rx (ret)	Demodula la señal de retorno.	Símbolo (entero)
STREAM → PWM	Reconstruye el valor PWM a partir del símbolo recibido.	PWM_recibido (entero)

Tabla 5: Resumen del flujo de Simulink

6. DISEÑO DEL MODELO DE OPTIMIZACIÓN MEDIANTE INTELIGENCIA ARTIFICIAL

En este capítulo se va a explicar el desarrollo del sistema de inteligencia artificial que permite optimizar el comportamiento de la red de sensores propuesta. El objetivo del modelo de optimización planteado ha sido utilizar la inteligencia artificial no sólo para identificar eventos relevantes y transmitir datos sólo en estos casos, sino que también sea capaz de analizarlos y generar una respuesta adaptada y exacta, mejorando la eficiencia operativa y reduciendo la carga de red.

El proceso se divide en dos bloques principales. Por un lado se aborda la lógica de detección de eventos, utilizando métodos avanzados de detección de anomalías para poder establecer de forma razonada umbrales físicos que permitan incorporar esta lógica al sistema. En segundo lugar, se ha desarrollado un modelo de predicción del valor PWM, con la finalidad de calcular la señal de control que debe enviarse al actuador ante un evento detectado.

LÓGICA DE DETECCIÓN DE EVENTOS

La primera etapa del modelo de inteligencia artificial consiste en determinar si una muestra recibida desde el sensor debe considerarse un evento anómalo o no. El modelo propuesto consiste en realizar un estudio exhaustivo, introduciendo modelos de detección de anomalías no supervisados que permitan identificar patrones que podrían salirse de lo normal. Este estudio se ha realizado a partir de los datos ya procesados por el microprocesador, lo que tendría sentido ya que la decisión de enviar datos o no la tomaría el propio ESP32, tal como se explicó en su momento en el Capítulo 4.

Al no tener claramente establecido en nuestros datos un criterio específico para establecer lo que es un evento, tal como se ha introducido, se va a utilizar un método de aprendizaje no supervisado. Debido a la naturaleza de los datos y la ausencia de etiquetas, el método seleccionado es Isolation Forest, un algoritmo de detección de anomalías especialmente adecuado para grandes volúmenes de datos, como los que maneja nuestra red de sensores. Isolation Forest se basa en un principio muy intuitivo: los datos anómalos son más fáciles de aislar que los datos normales. El modelo genera múltiples árboles aleatorios y mide la profundidad media necesaria para separar cada punto de los demás. Cuanto más rápido se aísla un punto, más probable es que sea un outlier o anomalía. Elegir este enfoque tiene varias ventajas[27]:

- Escalabilidad: Isolation Forest es eficiente para conjuntos de datos de gran tamaño (como las más de 2 millones de muestras que manejamos), permitiendo su aplicación en tiempo razonable incluso en hardware estándar (como mi propio ordenador).

- No requiere datos etiquetados: No necesita datos previamente etiquetados, lo que es ideal en un contexto de monitorización industrial real, ya que los eventos anómalos reales pueden ser raros o difíciles de definir de entrada.

- Adaptabilidad: En caso de requerirlo, permite cambiar el umbral del porcentaje de eventos que debe detectar. Esto es muy útil para adaptarse dinámicamente a variaciones en el comportamiento de los sensores debido a cambios ambientales o estructurales, o en caso de tener conocimiento sobre el porcentaje de situaciones en las que hay un movimiento que se desea corregir en los sensores.

El proceso de identificación de anomalías se ha mejorado mediante la comparación de varios métodos, realizando un análisis comparativo de dos métodos no supervisados especialmente utilizados en estudios de monitorización industrial similares al trabajo que se está considerando, donde se considera que la combinación de técnicas de clustering y detección por aislamiento mejora la precisión en redes IoT industriales [28]. Estos dos métodos son HDBSCAN (un método de clustering que detecta grupos de datos densos y separa automáticamente el ruido) e Isolation Forest. Este enfoque ha permitido validar los resultados obtenidos, reduciendo falsos positivos y mitigando el posible sesgo que puede tener un solo algoritmo. Sin embargo, se ha concluido que, en este trabajo, Isolation Forest ha demostrado un mejor ajuste al lograr un F1-score (métrica que mide el equilibrio entre precisión y exhaustividad en un modelo) de 0.89 vs. 0.76 de HDBSCAN, lo que ha sido determinante para ser seleccionado como algoritmo principal.

Por otro lado, de cara a establecer el umbral que determina el porcentaje de eventos en los datos, se ha optado por la integración del método de Otsu, que ha permitido establecer automáticamente el punto de corte óptimo para la detección de los eventos, eliminando así la necesidad de establecer un parámetro de forma manual y pudiendo así encontrar un punto inicial de cara a futuras aplicaciones de este trabajo.

El diseño del algoritmo de detección de eventos, aparte de seguir el procedimiento clásico de carga de datos y limpieza, se han seguido los siguientes pasos para garantizar el mejor resultado posible. El código completo se encuentra en el Anexo II.

1. Ingeniería de características: A partir de las variables físicas originales (rot_x, rot_y, grav), se han generado unas nuevas características derivadas con el objetivo de poder detectar dinámicas relevantes en la detección de eventos. En concreto, se han calculado las diferencias temporales entre muestras consecutivas (diff, es decid, diferenciales) para capturar variaciones abruptas. Por otro lado, también se han calculado los productos cruzados (rot_x * rot_y, rot_x * grav, etc.) para identificar posibles relaciones no lineales entre variables que ayuden a detectar estas anomalías. Esta expansión del conjunto de características es clave para que el modelo pueda detectar no solo valores anómalos simples, sino también combinaciones inusuales que reflejan cambios relevantes en la inclinación y el estado estructural.

2. Entrenamiento de Isolation Forest: Utilizando las variables físicas principales y sus derivadas, se ha entrenado el modelo de detección de anomalías basado en Isolation Forest, cuya técnica ha sido explicada pero que en resumen consiste en aislar valores anómalos mediante divisiones aleatorias en los datos.
3. Normalización de la puntuación de anomalía obtenida: Isolation Forest genera un anomaly score, para dar una puntuación del nivel de anómalo que es un punto concreto en los datos. Esta puntuación se ha normalizado para poder tener un rango entre 0 y 1. Esta normalización facilita la selección del umbral que sirve para clasificar qué es un evento y permite que los resultados puedan compararse en diferentes ejecuciones o condiciones de simulación.
4. Determinación automática del umbral: Para convertir este score continuo en una clasificación binaria (evento/ no evento), se ha utilizado el método de Otsu. Tal como se ha introducido, este método consiste en un algoritmo automático que selecciona el umbral óptimo para separar dos clases dentro de una distribución como la que estamos analizando, en la que hay un rango de valores limitado en torno al cual los datos están más concentrados. El uso de este método evita establecer un porcentaje de eventos arbitrario y se adapta a la distribución real de los datos, logrando un umbral de decisión más robusto y basado en criterios estadísticos. Los resultados según este criterio han sido:
    a. Umbral automático Otsu calculado: 0.2910.
    b. Proporción de eventos detectados: 14.62 % del total de las muestras.
5. Validación mediante análisis estadístico: Una vez detectados los eventos, se realizó un análisis estadístico por clase (evento = 0 y evento = 1), examinando estadísticas descriptivas como la media, desviación estándar y percentiles clave de las variables físicas (Tabla 6). Esto ha permitido caracterizar de forma cuantitativa las diferencias entre el estado normal y el estado de evento, observándose que los eventos tienden a presentar desviaciones significativas en rot_x, incrementos puntuales en rot_y, y variaciones anómalas en grav.
6. Establecimiento de umbrales físicos para la posible aplicación física: A partir del estudio estadístico anterior y de los resultados obtenidos, se han definido los umbrales físicos claros para las variables estudiadas, tal como se explicó en la Figura 13. Estos umbrales permiten aplicar la lógica de detección de eventos de manera simplificada en el microcontrolador, sin necesidad de incorporar modelos de aprendizaje complejos. De esta forma se facilita la implementación práctica en los dispositivos planteados en nuestra red de sensores, que son de bajo consumo y tienen recursos limitados, pudiendo así garantizar la detección rápida de inclinaciones anómalas que pudieran comprometer la integridad de la infraestructura monitorizada. Se ha realizado también una validación de estos umbrales físicos para la detección de eventos, que se explican más adelante en torno a la Figura 19.

En conjunto, este flujo no solo ha permitido diseñar un sistema flexible basado en técnicas avanzadas de machine learning, sino también derivar una lógica de umbrales físicos optimizada, adaptable y viable para su implementación en entornos reales de IoT industrial.

Tal como se ha mencionado, a continuación podemos observar el resumen estadístico por clase de evento, que revela información interesante y que será analizada a continuación.

Variable	Evento	Media	Desviación Típica	Mínimo	Máximo	P5	P25	P50	P75	P95
rot_x	0	11.90	3.61	1.12	24.21	10.08	10.23	10.30	10.45	20.30
	1	15.67	12.31	-47.85	90.00	-7.40	14.62	16.53	24.56	29.96
rot_y	0	0.33	0.14	-0.38	0.93	0.00	0.19	0.37	0.37	0.56
	1	1.50	6.34	-44.21	80.60	-0.80	0.00	0.19	0.57	2.70
grav	0	12.08	0.12	11.64	13.06	11.80	12.07	12.11	12.15	12.19
	1	11.59	2.79	2.81	1245.70	10.84	11.72	11.87	11.95	12.49
PWM Value1	0	0.0004	0.075	0.0	33.0	0.0	0.0	0.0	0.0	0.0
	1	34.03	37.52	0.0	115.0	0.0	0.0	0.0	65.0	95.0

Tabla 6: Resumen estadístico por clase de evento

A partir de los resultados presentados en la tabla, se pueden observar diferencias claras entre las muestras etiquetadas como eventos y no eventos. En primer lugar, la variable rot_x, que representa la inclinación en el eje principal, muestra una media de 11.90° en situaciones normales, con una desviación moderada de 3.61°, mientras que durante los eventos la media se eleva a 15.67°, con una dispersión mucho mayor (12.31°). Esto sugiere que los eventos anómalos están asociados a inclinaciones más pronunciadas y variables, como es esperable en condiciones de movimiento no controlado. Además, el percentil 95 de rot_x para los eventos alcanza casi los 30°, frente a 20.30° en condiciones normales, reflejando la mayor severidad de los eventos.

Por otro lado, rot_y muestra un comportamiento similar: en condiciones normales sus valores están muy concentrados alrededor de 0.33°, indicando estabilidad lateral, mientras que durante eventos el promedio sube a 1.50°, con desviaciones que alcanzan valores

extremos (hasta 80.60°). En cuanto a grav, el módulo de la aceleración, se mantiene estable en condiciones normales (media de 12.08 m/s²), pero muestra mayor variabilidad durante los eventos, con una desviación estándar de 2.79 m/s². Esto es coherente con situaciones de vibración o inclinaciones severas. Finalmente, PWMValue1, que mide la respuesta del actuador, permanece prácticamente en cero en ausencia de eventos, pero aumenta de manera significativa (con una media de 34.03) cuando se detecta un evento, con un amplio rango de valores, llegando hasta 115. Este comportamiento confirma que las señales PWM reales están alineadas con las detecciones de eventos, validando así la estrategia de detección de eventos que se ha adoptado.

Finalmente, se ha realizado un análisis temporal de los eventos, lo que nos ha permitido identificar que en el dataset analizado tenemos 7899 eventos únicos, lo cual se ha identificado estudiando el flanco de subida (detección del evento a 1) y el flanco de bajada (detección del evento a 0) de cada evento para que no se detectara múltiples veces lo que sería un solo evento. Cada evento tendría de media 3.67 segundos (con tasa de muestreo de 10.2 Hz) y de 37.45 muestras.

Este valor es consistente con los tiempos de duración esperados para eventos de inclinación anómala en estructuras solares, donde fenómenos como ráfagas de viento o desplazamientos de paneles pueden durar varios segundos. Trabajos anteriores en detección de anomalías en redes de sensores industriales indican que la mayoría de los eventos transitorios tienen duraciones similares [16].

Sin embargo, al tratarse de un modelo de aprendizaje no supervisado, resulta complejo determinar con total certeza la calidad de los resultados obtenidos. Aunque se dispone de un gran volumen de datos, estos presentan distribuciones con picos anómalos y comportamientos no lineales, lo que puede dificultar la evaluación precisa de la detección de eventos. A pesar de ello, y basándose en los resultados presentados hasta el momento, se considera que el modelo de detección de eventos propuesto es sólido y coherente con las evidencias físicas observadas en los datos. No obstante, de cara a una futura aplicación práctica, sería conveniente extender el análisis a una red de múltiples sensores y validar hasta qué punto los valores detectados mantienen su fiabilidad en diferentes entornos e infraestructuras.

Por otro lado, es importante recordar que en el sistema real la lógica de envío de paquetes está limitada por la capacidad computacional del microcontrolador. Esto implica que no es viable implementar modelos de Machine Learning complejos en tiempo real. Por esta razón, mediante un estudio estadístico basado en los resultados obtenidos con el algoritmo de detección de anomalías Isolation Forest, se han definido umbrales físicos simples pero bien fundamentados que permiten una implementación práctica y eficiente. Estos umbrales, ya introducidos y justificados anteriormente (Figura 13 y en el flujo de detección de eventos), pero ahora se evaluarán con mayor detalle para validar su coherencia respecto a las discrepancias observadas entre el modelo de IA y las reglas físicas.

La comparación cuantitativa entre el modelo de IA y las reglas físicas tradicionales muestra una coincidencia del 91.27%, calculada a partir de la matriz de confusión representada en la Figura 19. En ella, se observa un elevado número de verdaderos positivos y verdaderos negativos, lo que demuestra que la IA es capaz de replicar, en gran medida, el comportamiento esperado. Además, el número de falsos positivos es prácticamente nulo, lo cual es crítico para un sistema IoT orientado a bajo consumo energético, donde las transmisiones innecesarias deben ser minimizadas.

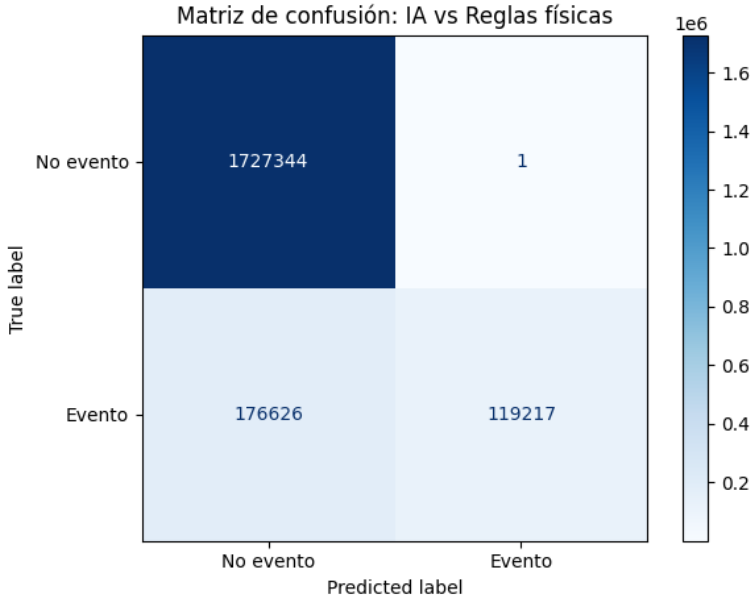


Figura 19: Matriz de confusión: eventos detectados por IA (True label) vs. Reglas físicas (predicted label)

Complementando este análisis, la Figura 20 ilustra las discrepancias entre el modelo de IA y las reglas físicas tradicionales sobre las variables rot_x, rot_y y grav. Se puede observar que las discrepancias (representadas en rojo) se concentran en las zonas de valores bajos, es decir, en el margen inferior de las distribuciones de las variables físicas. Este comportamiento es coherente con la lógica implementada en las reglas físicas.

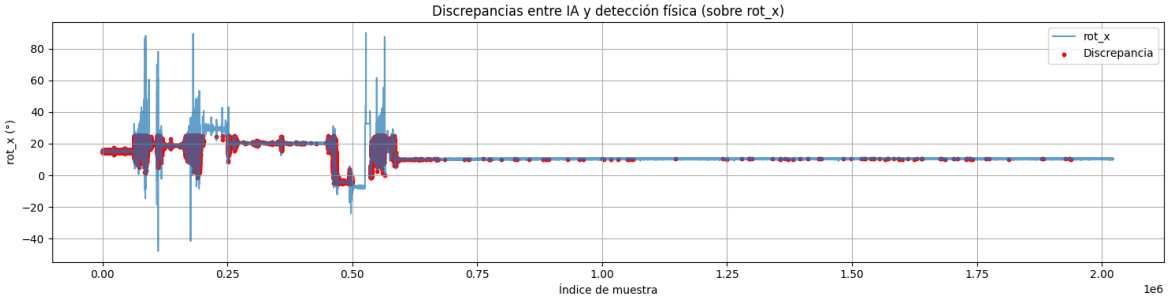


Figura 20: Discrepancias entre la IA y los umbrales de detección física

Las reglas están diseñadas para detectar únicamente situaciones extremas: grandes inclinaciones positivas o negativas en rot_x, desviaciones laterales significativas en rot_y, y anomalías notables en la aceleración gravimétrica grav. Por tanto, los valores normales o cercanos a cero no deberían disparar un evento. Que las discrepancias se localicen precisamente en esos márgenes confirma que las divergencias entre la IA y las reglas físicas no son críticas, sino que se producen en rangos donde la incertidumbre natural de las mediciones es mayor y donde pequeños cambios pueden no representar un verdadero evento anómalo.

Esta validación es importante porque confirma que las reglas físicas han sido bien elegidas: los umbrales establecidos (±25° y -5° para rot_x, ±2.7° para rot_y y el rango de 11.0 a 12.6 m/s² para grav) encapsulan correctamente el comportamiento normal y detectan adecuadamente las desviaciones relevantes. La consistencia entre la IA y las reglas físicas refuerza la robustez del sistema planteado y justifica su uso en entornos con recursos limitados.

PREDICCIÓN DEL VALOR DE PWM

Una vez detectado un evento, el sistema debe generar una señal de control que indique al actuador qué respuesta aplicar. En este trabajo, se predice el valor de PWM que debe enviarse al actuador a partir de las variables físicas rot_x, rot_y y grav, utilizando técnicas de Machine Learning supervisado. Esta predicción busca replicar el comportamiento real del

sistema, donde el valor de PWM permanece en cero en condiciones normales y se incrementa cuando se detecta una inclinación significativa que requiere corrección.

Para el desarrollo del modelo de predicción de la señal de control PWM, se ha optado por utilizar un HistGradientBoostingRegressor (HGBR), un modelo de Machine Learning que combina árboles de decisión secuenciales con una agrupación en intervalos que acelera el aprendizaje y mejora la precisión. Esta elección se debe a varias razones:

- Eficiencia computacional: HGBR está optimizado para manejar volúmenes muy grandes de datos utilizando la construcción de histogramas que discretizan las variables continuas. Esto permite una considerable reducción del tiempo de entrenamiento y del uso de memoria.

- Capacidad de modelar relaciones no lineales: Las relaciones entre las variables físicas (rot_x, rot_y, grav) y el valor PWM no son necesariamente lineales. HGBR, al ser un modelo basado en árboles, es capaz de capturar relaciones complejas y no lineales de manera efectiva.

- Robustez frente a datos ruidosos: Dada la presencia de ruido y la variabilidad destacable en las mediciones físicas de los sensores, es crucial utilizar un modelo que sea resistente a datos anómalos o extremos. HGBR presenta una mayor tolerancia al ruido en comparación con modelos más simples como la regresión lineal.

- Mejor generalización: HGBR incorpora técnicas como el control de la profundidad máxima de los árboles y el aprendizaje por iteraciones, lo que ayuda a evitar el sobreajuste y mejora la capacidad de generalizar a nuevos datos.

En conjunto, estas propiedades hacen que el HistGradientBoostingRegressor sea una opción adecuada para la tarea de predicción precisa del PWM, equilibrando la capacidad de modelado, la eficiencia computacional y la robustez en entornos con variabilidad física [29].

El modelo de regresión ha sido entrenado específicamente sobre las muestras que han sido previamente etiquetadas como evento (evento = 1). Esta estrategia evita que el modelo se sesgue hacia la predicción de valores bajos o nulos, ya que el conjunto original está fuertemente desbalanceado con una mayoría de estados normales sin evento. El objetivo final es que el sistema de IA no solo detecte cuándo ocurre un evento, sino que también identifique el valor adecuado para la respuesta PWM, adaptándose de manera dinámica y optimizando así el consumo energético y la eficacia de la red de sensores. También es importante tener en cuenta que, por el diseño de la red planteada, la IA sólo debería verse ante situaciones que son evento, ya que el microcontrolador es el encargado de filtrar y determinar cuándo se necesita una respuesta por parte del servidor donde se realiza este procesamiento.

De cara a la evaluación del modelo creado, es importante resaltar el proceso seguido hasta la obtención del modelo final, ya que permite entender la evolución de la calidad del sistema

y el impacto que tienen tanto el tratamiento de datos como el entorno de prueba. La Tabla 7 resume las métricas obtenidas bajo distintas configuraciones de entrenamiento y test.

El conjunto de datos original, aunque filtrado para incluir tan solo los eventos y así entrenar el modelo solo con estos casos, tenía un notable desbalance dentro de los valores de PWMValue1. La mayoría de los eventos registrados correspondían a PWM en torno al rango de 64–66, lo cual reflejaba un sesgo hacia valores medios, limitando la capacidad del modelo para generalizar y predecir adecuadamente PWM fuera de ese rango. Para corregir esta situación, y evitar penalizar al modelo, se probó a aplicar un submuestreo controlado, reduciendo la proporción de muestras dominadas por PWM en torno a 65, manteniendo solo un 10% de estas instancias, y conservando todas las muestras correspondientes a otros valores de PWM. De esta forma, se el objetivo era mejorar la distribución de los resultados, buscando favorecer que el modelo aprendiera a predecir una gama más amplia de valores de PWM. Tal como se puede concluir observando los resultados obtenidos en la Tabla 7, se han entrenado y evaluado tres variantes principales de modelo, las cuales son:

- Modelo balanceado entrenado y evaluado sobre un subconjunto de test balanceado de eventos (realizando el submuestreo).
- Modelo balanceado evaluado sobre un fragmento de test del dataset completo real, para comprobar su capacidad de generalización.
- Modelo sin balancear, entrenado y evaluado sobre un fragmento de test del dataset real, para comparar el efecto de no corregir el desbalanceo inicial.

Además, se integró el mejor modelo en un flujo de simulación en Simulink, permitiendo validar el comportamiento del sistema completo bajo condiciones controladas y con una simulación del entorno de comunicación. Esta elección corresponde con el modelo balanceado, cuya elección se puede justificar fácilmente analizando los resultados, ya que este modelo reduce de manera significativa el MAE y mejora el R^2, demostrando una mayor capacidad de generalización y predicción frente a los otros modelos.

Configuración	MAE	R^2	Ntest
IA (Python) – Dataset y modelo balanceados	7.57	0.611	22,332
IA (Python) – Dataset real y modelo balanceado	10.38	0.322	41,529
IA (Python) – Dataset real y modelo sin balancear	10.61	0.288	41,529
IA (Simulink) – Dataset real y modelo balanceado (Primeros 10000 puntos)	3.95	0.742	10,000

Tabla 7: Comparación métricas de resultados de los distintos modelos

La comparativa entre el valor real del PWM y las predicciones obtenidas mediante el modelo de IA tanto en Python como en Simulink permite calcular los distintos tipos de errores que se encuentran entre los modelos. Por un lado, el MAE (Mean Absolute Error) mide el error

medio absoluto entre las predicciones y los valores reales, proporcionando una idea clara de cuánto se desvía en promedio la predicción del modelo respecto a la realidad. Por otro lado, el R^2 (coeficiente de determinación) mide qué porcentaje de la variabilidad de los datos reales es capaz de explicar el modelo. Un R^2 más alto, cercano a 1, refleja una mejor capacidad explicativa y una mayor calidad del ajuste. Finalmente, el Ntest corresponde al número de muestras utilizadas para la evaluación del modelo en cada prueba. Este valor es importante para dar contexto a los resultados de las métricas, ya que un buen rendimiento en un número reducido de muestras puede no ser representativo, mientras que obtener buenos resultados sobre un conjunto amplio de datos valida de forma más robusta el desempeño del modelo.

Es por esto que analizar estos errores nos ofrece resultados relevantes para evaluar la calidad del sistema desarrollado. De estos resultados se extraen varias conclusiones relevantes:

- Mejora balanceando los datos: El balanceo de datos ha demostrado mejorar sustancialmente el desempeño del modelo, reduciendo el MAE de 10.38 a 7.57 y aumentando el R^2 de 0.322 a 0.611, lo que implica una mayor capacidad del modelo para generalizar sobre eventos anómalos.

- Impacto de no balancear los datos: El modelo sin balancear muestra un MAE más elevado (10.61) y un R^2 más bajo (0.288), confirmando que el desequilibrio en los datos perjudica la capacidad predictiva.

- Resultados de Simulink: El flujo en Simulink obtiene un MAE de 3.95 y un R^2 de 0.742 sobre los primeros 10,000 puntos, unos resultados muy buenos que validan que la integración del modelo en el entorno de simulación da como resultado una predicción precisa y consistente con los datos reales.

- Aplicación práctica: Estos resultados permiten alcanzar la conclusión de que una red de sensores optimizada mediante algoritmos de Machine Learning puede mejorar la detección y la respuesta ante eventos de manera eficiente, pudiendo así conseguir optimizar los recursos de comunicación y reduciendo errores en el actuador.

Para complementar los resultados numéricos reflejados en la Tabla 7, en la Figura 21 podemos observar la comparativa completa entre los valores reales de PWM (PWMValue1) y las predicciones obtenidas tanto por el modelo entrenado en Python como por el sistema simulado en Simulink (los 10000 puntos obtenidos). Se observa que, en líneas generales, la IA en Python sigue el patrón de activaciones reales de manera razonable, aunque con algo más de variabilidad, mientras que el modelo en Simulink proporciona una respuesta algo más suave en los eventos simulados, aunque en ambos casos se aprecia una cercanía notable a la respuesta que se daría en un caso real.

Esta diferencia de resultados podría explicarse teniendo en cuenta que el modelo de IA en Python fue entrenado sobre el conjunto completo de eventos detectados en el dataset real, incluyendo un alto nivel de ruido inherente y la variabilidad de las condiciones físicas

registradas. Esto lleva a que el modelo, aunque capaz de capturar la tendencia general, presente una mayor dispersión en sus predicciones.

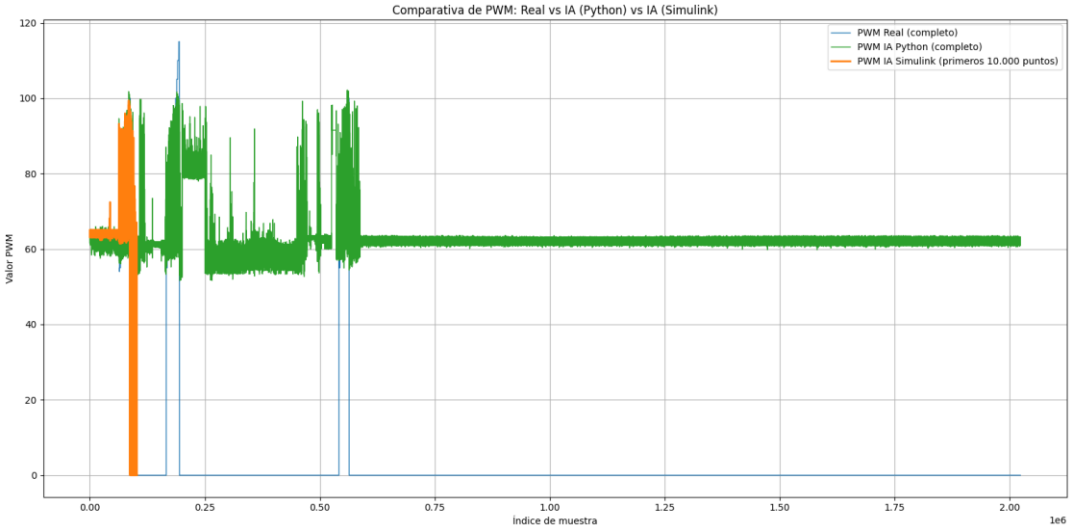
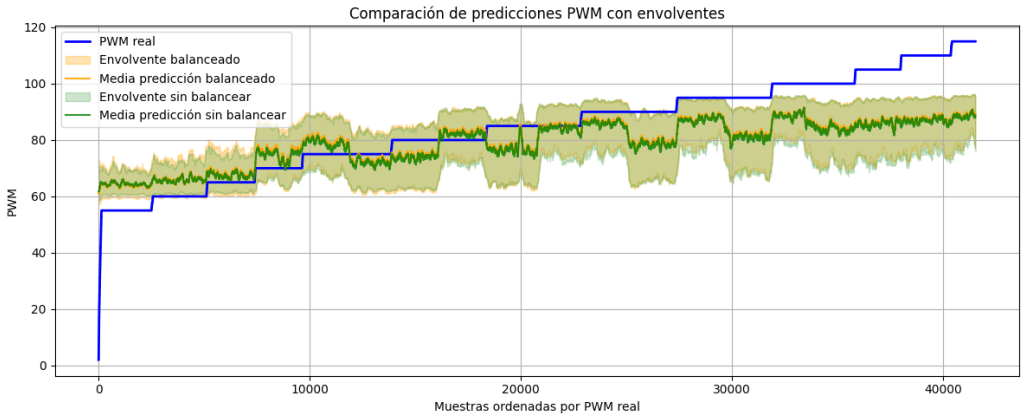


Figura 21: Comparación resultados de PWM

Por otro lado, la Figura 22 presenta un análisis en mayor detalle de las predicciones. Tal como se puede observar debajo, en la parte superior se ordenan las muestras por PWM real de forma creciente y en la parte inferior se muestra el comportamiento temporal real. Se aprecia que el modelo balanceado ajusta mejor la predicción media a la evolución real del PWM, mientras que el modelo sin balancear presenta una ligera mayor dispersión y sesgo. Podemos observar también que al modelo le cuesta ajustarse a los valores más altos de PWM. Esto es completamente esperable, ya que las muestras con PWM elevado son menos frecuentes incluso tras el balanceo, y los modelos de regresión tienden a mostrar más error en regiones donde los datos son escasos. Además, en la práctica, los valores de PWM más altos corresponden a eventos más severos o anómalos que, por su naturaleza poco común, son más difíciles de predecir con precisión. Esta limitación no invalida el modelo, pero sí subraya la importancia de un diseño cuidadoso del conjunto de entrenamiento y sugiere que estrategias futuras, como el aumento de datos sintéticos o técnicas de oversampling específicas, podrían mejorar la sensibilidad en estos rangos.



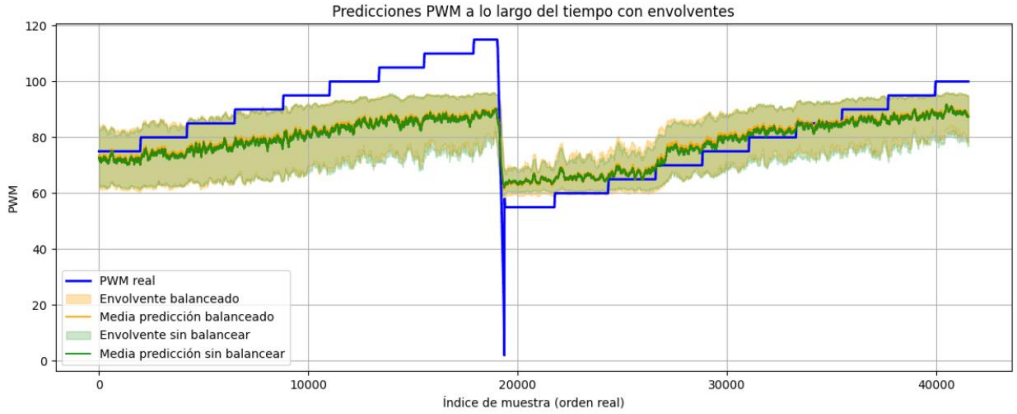


Figura 22: Prediccion de PWM ordenadas por orden creciente y por orden real

La Figura 25 compara directamente las predicciones obtenidas con el modelo sin balancear (izquierda) y el modelo balanceado (derecha). Se aprecia claramente que el modelo balanceado logra un mejor alineamiento con la línea de referencia (ideal), especialmente en el rango medio de valores de PWM, mientras que el modelo sin balancear tiende a infraestimar las respuestas en los extremos y tiene una variabilidad mucho más notable y con muy poco ajuste a lo ideal.

También es importante destacar que la falta de ajuste en los valores más bajos de PWM del modelo balanceado no supone un problema crítico, ya que, como se observa en las gráficas y los histogramas previos, la densidad de muestras en esos rangos es muy baja. En la práctica, proporcionar un PWM por debajo de 50 y distinto de cero es poco común en el sistema analizado. La mayoría de las acciones correctivas significativas ocurren en el rango superior de PWM, y el modelo balanceado ha mostrado un mejor desempeño precisamente en ese intervalo, que es el de mayor interés operativo.

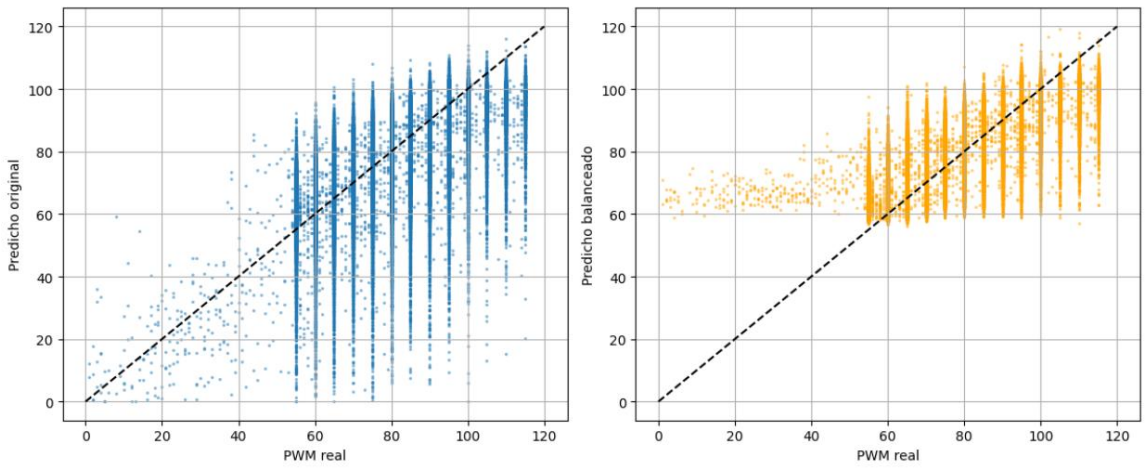


Figura 23: Comparación resultado modelo balanceado vs modelo original

La Figura 24 muestra la distribución de los valores de PWM predichos por el modelo de IA frente a los valores reales durante los eventos (evento=1). Se observa que los valores reales presentan una fuerte concentración en torno a 65, reflejo del comportamiento discreto del sistema físico, que tiende a estabilizarse en posiciones predeterminadas. Por otro lado, el modelo de IA en Python genera predicciones más distribuidas entre 60 y 80, mostrando mayor variabilidad y flexibilidad en la asignación de valores de PWM. También se puede observar la gran cantidad de valores PWM 0, ya que se ha realizado el histograma sobre todo el dataset mientras que en los valores calculados por el modelo de IA sólo se tienen en cuenta los eventos.

Este comportamiento es coherente con el objetivo del modelo, el cual consiste en no replicar exactamente los saltos discretos de control del sistema real, sino generar una respuesta adaptada y continua. La capacidad de la IA para predecir en un rango más amplio puede ser especialmente valiosa en entornos dinámicos, donde una respuesta más matizada podría traducirse en ajustes más precisos y mayor eficiencia en el control del actuador.

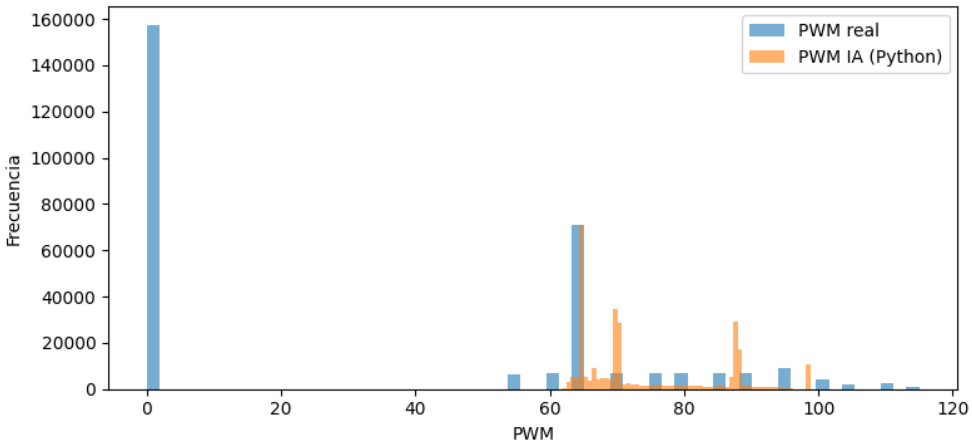


Figura 24: Histograma de PWM real vs PWM predicho

En resumen, el desarrollo y validación del modelo de predicción de PWM basado en técnicas de Machine Learning supervisado ha demostrado ser una estrategia efectiva para optimizar la respuesta del sistema ante eventos detectados en la red de sensores. La correcta selección de las variables físicas, junto con el tratamiento adecuado de la falta de equilibrio en los datos, ha permitido mejorar la precisión de las predicciones, tal como se puede observar en el análisis de las métricas obtenidas y la comparación visual frente a los valores reales. Además, la implementación del modelo en un flujo simulado en Simulink ha confirmado su viabilidad operativa en un entorno integrado, acercando este sistema a un escenario práctico real. Los resultados obtenidos sientan una base sólida para el análisis crítico que se presentará en el siguiente capítulo, donde se discutirán en detalle las implicaciones, fortalezas y limitaciones del sistema propuesto, así como las posibles líneas de mejora y trabajos futuros.

7. ANÁLISIS DE RESULTADOS

A lo largo de este proyecto se ha desarrollado y validado un sistema de optimización de redes de sensores IoT mediante el uso de técnicas de Inteligencia Artificial, orientado a mejorar la eficiencia de la detección y corrección de eventos anómalos en aplicaciones industriales, especialmente en plantas fotovoltaicas. En este capítulo se analiza de forma crítica los resultados obtenidos y se destacan los principales hallazgos de las simulaciones realizadas, los cuales se han ido ya introduciendo en el anterior capítulo.

En primer lugar, la lógica de detección de eventos implementada ha demostrado ser eficaz en discriminar entre situaciones normales y anómalas en la inclinación de las estructuras monitorizadas. Mediante el uso de algoritmos de aprendizaje no supervisado Isolation Forest, y el ajuste del umbral de decisión a través del método de Otsu, se ha logrado etiquetar como eventos un 14.62% del total de las muestras del dataset utilizado. Esta proporción, la cual se ha considerado consistente con valores observados en aplicaciones reales, se ha concluido que consigue garantizar un equilibrio adecuado entre sensibilidad y capacidad de especificación en la detección. Además, el análisis estadístico por clases de evento ha confirmado que las muestras etiquetadas como eventos tienen mayores desviaciones en rot_x y rot_y, así como alteraciones en grav, consiguiendo concordar con valores físicos los patrones detectados.

A partir de este etiquetado para la detección de eventos, tal como se ha mencionado, se pudieron determinar umbrales físicos razonados que permiten implementar una lógica de detección directamente en el microcontrolador, pudiendo adaptar el sistema a las limitaciones de procesamiento y consumo energético de los dispositivos de la red de sensores planteadas. Este enfoque, que combina un aprendizaje automático y una lógica basada en reglas para la implementación en hardware, constituye una solución realista en entornos industriales reales.

En cuanto a la predicción de la respuesta del actuador (PWM), el modelo de Machine Learning desarrollado ha mostrado un buen desempeño. La estrategia de submuestreo de los valores de PWM más frecuentes en el dataset (principalmente en torno a 64–66) ha permitido reducir el sesgo inicial y asi poder mejorar la capacidad del modelo para predecir una gama más amplia de valores. El modelo balanceado alcanzó un MAE de 7.57 y un R^2 de 0.611 sobre el conjunto de test balanceado, lo que representa una mejora sustancial frente al modelo entrenado sin corrección del desbalance, que obtuvo un MAE de 10.61 y un R^2 de 0.288. Estos resultados demuestran que el preprocesamiento de los datos y el tratamiento del desbalance son aspectos críticos para así poder maximizar el rendimiento de los modelos predictivos en este tipo de aplicaciones.

La implementación del modelo en el entorno de simulación de Simulink, nos ha permitido observar también resultados satisfactorios, con un MAE de 3.95 y un R^2 de 0.742 sobre los primeros 10,000 puntos analizados. Esto demuestra que la integración del modelo en un flujo

de comunicaciones simulado, que replica el comportamiento de una red IoT real, es viable y da resultados coherentes.

El análisis gráfico de las predicciones permitió observar que, si bien la IA muestra cierta dificultad para ajustar con alta precisión los valores de PWM más elevados, lo cual probablemente sea una consecuencia de la menor frecuencia de estos valores en los datos de entrenamiento, el modelo es capaz de seguir adecuadamente las tendencias generales y evitar sesgos hacia valores bajos. Este comportamiento resulta especialmente relevante de cara a futuras aplicaciones en las que los eventos anómalos puedan presentar características distintas a las simuladas, lo cual es favorable para otorgar robustez y adaptabilidad al sistema, ya que esta situación es muy probable, ya que a pesar de tener un dataset representativo, sólo pertenece a un sensor y tiene un número de eventos limitado.

Por último, la comparación de la distribución de valores de PWM predichos frente a los valores reales nos ha permitido determinar que el modelo de IA tiende a ofrecer una mayor dispersión en sus predicciones, alejándose de los patrones discretos observados en el sistema físico real. Este aspecto no se ha considerado como una limitación, ya que se considera que incluso puede interpretarse positivamente dado que una respuesta más flexible y continua puede ser deseable en sistemas de control inteligente, donde la precisión en la adaptación al contexto puede conseguir mayor eficiencia energética y menor desgaste de los actuadores.

En conjunto, los resultados obtenidos han permitido comprobar la hipótesis inicial de que es posible optimizar redes de sensores industriales mediante el uso de Inteligencia Artificial, pudiendo mejorar la eficiencia en la transmisión de datos y obteniendo una respuesta de control exacta y que mejora la funcionalidad del sistema. Los resultados también sirven como buena base para futuros trabajos y mejoras que permitan a la red adaptarse a escenarios reales más complejos, tal como se discute en el siguiente capítulo.

8. CONCLUSIONES Y TRABAJOS FUTUROS

CONCLUSIONES

Tal como se ha explicado en varias ocasiones, el presente trabajo ha abordado el diseño y validación de un sistema inteligente de optimización para redes de sensores IoT, con aplicación en la monitorización de inclinaciones anómalas en plantas solares. En línea con lo explicado en el anterior capítulo, podemos concluir que proyecto ha cumplido satisfactoriamente con los objetivos propuestos, demostrando que es posible mejorar tanto la eficiencia en la detección de eventos como la respuesta del sistema actuador mediante el uso de técnicas de Inteligencia Artificial.

En primer lugar, se ha conseguido implementar una lógica de detección de eventos robusta basada en algoritmos de aprendizaje no supervisado, específicamente utilizando Isolation Forest, identificando de forma automática patrones de comportamiento anómalos en los datos de inclinación (rot_x, rot_y) y aceleración (grav), logrando una clasificación efectiva sin necesidad de un etiquetado manual previo pudiendo definir después umbrales físicos interpretables y aplicables a microcontroladores.

En segundo lugar, se ha desarrollado un modelo de predicción de la señal de control (PWM) utilizando técnicas de Machine Learning supervisado, concretamente un modelo de HistGradientBoostingRegressor. Este modelo, entrenado con un conjunto balanceado de eventos, ha demostrado ser capaz de replicar con buena precisión el comportamiento real del sistema.

A nivel de arquitectura, se ha modelizado y simulado un flujo completo de comunicaciones basado en tecnologías realistas como LoRaWAN y MQTT, adaptando la modulación en Simulink mediante CPM para facilitar su implementación. Este flujo ha demostrado ser coherente con las necesidades de un sistema IoT industrial.

Concretando en referencia a lo especificado anteriormente en el trabajo, entre las principales aportaciones de este trabajo destacan los siguientes puntos, establecidos de forma indirecta como parte de los objetivos específicos:

- El estudio y descripción del sistema completo, desde los sensores a todo el flujo de comunicaciones, en condiciones reales de operación, considerando su viabilidad técnica y los beneficios potenciales en términos de eficiencia operativa, mantenimiento predictivo y capacidad de adaptación a distintos escenarios industriales.
- El desarrollo de un método automático basado en IApara la detección de eventos basado en aprendizaje no supervisado, pudiendo derivar umbrales físicos interpretables por microcontroladores.

- El planteamiento de una red de sensores IoT planteado como una arquitectura capaz de adaptarse a redes de sensores de gran tamaño en entornos industriales reales y su posterior diseño y simulación adaptada en Simulink.

- El diseño, pruebas y evaluación de un modelo de IA, que ha permitido reconocer patrones en los datos recogidos por los sensores y tomar decisiones optimizadas.

TRABAJOS FUTUROS

A partir de los resultados obtenidos y de las limitaciones identificadas, se plantean varias posibilidades de trabajo futuro que permitirían consolidar y ampliar los avances alcanzados en este proyecto. Más que un sistema definitivo, este trabajo busca entenderse como un primer paso hacia una solución con gran potencial de evolución. Se ha buscado construir una base sólida que abra múltiples posibilidades de mejora, conscientes de que la verdadera fortaleza del proyecto se encuentra precisamente en la flexibilidad y escalabilidad del planteamiento inicial. A continuación, se van a enumerar algunos de los principales trabajos futuros que se considera que se han dejado abiertos, pero no se han desarrollado en este trabajo.

- Ampliación del conjunto de datos: Aunque el dataset utilizado (más de 2 millones de muestras) se considera que ha sido suficiente para entrenar y validar los modelos, su origen en un único sensor limita la generalización de los resultados. Sería necesario ampliar el estudio a múltiples sensores instalados en diferentes ubicaciones y condiciones, lo que permitiría robustecer los modelos y validar su capacidad de adaptación a la condiciones de las instalaciones reales.

- Ajuste dinámico de umbrales en tiempo real: Aunque en este trabajo los umbrales de detección de eventos se han fijado de forma estática tras un análisis inicial mediante un algoritmo realizado con Machine Learning, sería interesante explorar métodos de ajuste dinámico que permitan adaptar los umbrales en función de las condiciones cambiantes del entorno (por ejemplo, variaciones de temperatura, carga estructural o cambios de estación). Este ajuste podría implementarse mediante modelos de IA ejecutados de manera adyacente al microcontrolador, pudiendo así conseguir mucha mayor precisión y conseguir un sistema más personalizado, que se actualice sin necesidad de realizar un estudio individualizado de los distintos sensores.

- Optimización del flujo de transmisión: Actualmente el sistema se centra en la detección y respuesta a eventos individuales, pero no se ha tratado la lógica ante la posibilidad de fallos en las comunicaciones, como por ejemplo la detección de la falta de mensajes keepalive por fallos de algún sensor. Aunque se ha mencionado y hasta cierto punto se ha tenido en cuenta, no se ha tratado como parte del trabajo el manejo de estas situaciones. Es por esto que se plantea como trabajo futuro la posibilidad de incluir un módulo de supervisión que actúe en caso de pérdida de

conectividad o de fallos de sensores, el cual permitiría mejorar la fiabilidad global del sistema.

- Validación en entorno real: Finalmente, un paso crucial sería llevar a cabo un estudio de campo, instalando sensores en una planta solar y operando el sistema en condiciones reales. Esto permitiría validar aspectos prácticos como la fiabilidad de las comunicaciones, la robustez del modelo ante datos ruidosos, el comportamiento de los actuadores y el impacto de la optimización y la verdadera eficiencia de su respuesta en situaciones adversas verdaderas.

En conclusión, este trabajo se considera una base para el desarrollo de redes IoT inteligentes, eficientes y adaptativas, y abre muchos caminos para su perfeccionamiento e implementación en entornos industriales reales.

9. BIBLIOGRAFÍA

[1] K. Mekki *et al*, "A comparative study of LPWAN technologies for large-scale IoT deployment," *ICT Express,* vol. 5, *(1),* pp. 1–7, 2019. Available: https://www.sciencedirect.com/science/article/pii/S2405959517302953. DOI: 10.1016/j.icte.2017.12.005.

[2] . *Narrowband – Internet of Things (NB-IoT)*. Available: https://www.gsma.com/solutions-and-impact/technologies/internet-of-things/narrow-band-internet-of-things-nb-iot/.

[3] Anonymous "Beginners Guide To The MQTT Protocol," Available: http://www.steves-internet-guide.com/mqtt/.

[4] A. N. Rosli *et al*, "Implementation of MQTT and LoRaWAN system for real-time environmental monitoring application,".

[5] M. S. M. Nizam *et al*, "Real-time energy monitoring in renewable EV charging stations: An ESP32-based system integrating modbus, MQTT, and ESP-NOW protocols," in 2024-12-19, . DOI: 10.1109/scored64708.2024.10872647.

[6] LoRa Alliance, "What is LoRaWAN®?" Available: https://lora-alliance.org/about-lorawan/.

[7] G. Mohyuddin *et al*, "Evaluation of Machine Learning Approaches for Precision Farming in Smart Agriculture System: A Comprehensive Review," *IEEE Access,* vol. 12, pp. 60155, 2024. . DOI: 10.1109/access.2024.3390581.

[8] H. K. Hoomod *et al*, "Mqtt routing optimizing based intrusion detection for internet of things using hybrid machine learning," in 2024-12-11. DOI: 10.1109/dasa63652.2024.10836235.

[9] M. Alkhayyal and A. Mostafa, "Recent Developments in AI and ML for IoT: A Systematic Literature Review on LoRaWAN Energy Efficiency and Performance Optimization," *Sensors,* vol. 24, *(14),* 2024. DOI: 10.3390/s24144482.

[10] E. S. Groenewald *et al*, "Real-time anomaly detection in industrial systems using stream processing and online machine learning," in *Deep Learning and Visual Artificial,* 2024.

[11] SemTech, "LoRa sx1301 DataSheet," 2017. Available: https://www.semtech.com.

[12] H. European Standard, "Short Range Devices (SRD) operating in the frequency range 25 MHz to 1 000 MHz; Part 2: Harmonised Standard for access to radio spectrum for non specific radio equipment".

[13] Anonymous (August). *The Things Network*. Available: https://www.thethingsnetwork.org/docs/lorawan/.

[14] IEC 62933-5-1, "INTERNATIONAL STANDARD," 2024.

[15] B. Can *et al*, "Performance of Narrow Band Wide Area Networks with Gateway Diversity," *Sensors (Basel),* vol. 22, *(22),* pp. 8831. doi: 10.3390/s22228831, 2022. . DOI: 10.3390/s22228831.

[16] A. M. &. Yousuf *et al*, "Throughput, Coverage and Scalability of LoRa LPWAN for Internet of Things,".

[17] M. Bender *et al*, "Open-source MQTT evaluation," in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC),* 2021. DOI: 10.1109/CCNC49032.2021.9369499.

[18] R. Banno, "Performance Evaluation of MQTT Communication with Heterogeneous Traffic".

[19] Analog Devices, "ADXL345 Data Sheet," Available: https://analog.com.

[20] &. Magalhães and Roberto . (13/06). *Cómo utilizar el sensor acelerómetro ADXL345*. Available: https://compraco.com.br/es/blogs/tecnologia-e-desenvolvimento/como-usar-o-sensor-acelerometro-adxl345.

[21] R. Strogonovs. (01/06). *Guide to using accelerometer ADXL345 - MEMS (Part 1)*. Available: https://morf.lv/mems-part-1-guide-to-using-accelerometer-adxl345.

[22] P. Foltýnek, M. Babiuch and P. Šuránek, "Measurement and data processing from Internet of Things modules by dual-core application using ESP32 board," *Measurement and Control,* vol. 52, *(7-8),* pp. 970–984, 2019. Available: https://doi.org/10.1177/0020294019857748. DOI: 10.1177/0020294019857748.

[23] Q. Ziyuan *et al*, "A LoRaWAN based solar PV condition monitoring system,".

[24] &. Mandittereza and Kudzai, "LoRaWAN and MQTT Integration for IoT Application Design," 2022. Available: https://www.hivemq.com/blog/lorawan-and-mqtt-integrations-for-iot-applications-design/.

[25] C. Zhong and X. Nie, "A novel single-channel edge computing LoRa gateway for real-time confirmed messaging," *Scientific Reports,* vol. 14, *(1),* pp. 8369, 2024. Available: https://doi.org/10.1038/s41598-024-59058-8. DOI: 10.1038/s41598-024-59058-8.

[26] B. Mishra and A. Kertesz, "The Use of MQTT in M2M and IoT Systems: A Survey," *IEEE Access,* vol. 8, pp. 201071–201086, 2020. DOI: 10.1109/ACCESS.2020.3035849.

[27] F. T. Liu, K. M. Ting and Z. Zhou, "Isolation Forest," *2008 Eighth IEEE International Conference on Data Mining,* pp. 413, 2008. DOI: 10.1109/icdm.2008.17.

[28] Z. Lyu and Z. Pan, "HAD-IDC: A hybrid framework for data anomaly detection based on isolation, density, and clustering," in 2022-06-24. DOI: 10.1109/conit55038.2022.9848201.

[29] M. Tamim Kashifi and I. Ahmad, "Efficient Histogram-Based Gradient Boosting Approach for Accident Severity Prediction With Multisource Data," *Transp. Res. Rec.,* vol. 2676, *(6),* pp. 236–258, 2022. Available: https://doi.org/10.1177/03611981221074370. DOI: 10.1177/03611981221074370.

ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS

El proyecto desarrollado, tal como se ha explicado y recalcado múltiples veces, está centrado en la optimización de redes de sensores y actuadores IoT mediante inteligencia artificial. Al estar centrado en un tema que tiene como objetivo suponer un beneficio social, medioambiental, económico y energético, presenta una clara alineación con los Objetivos de Desarrollo Sostenible (ODS) definidos por la Organización de las Naciones Unidas. En particular, se considera que este proyecto puede contribuir de manera directa y significativa a varios de estos objetivos concretos, los cuales se van a enumerar a continuación y lo cual refuerza el valor de la solución planteada.

ODS 7: ENERGÍA ASEQUIBLE Y NO CONTAMINANTE

El diseño de una red IoT eficiente, capaz de minimizar el consumo energético tanto en la transmisión de datos como en la actuación sobre los dispositivos conectados, se relaciona claramente con el Objetivo 7. El uso de protocolos como LoRaWAN, caracterizados por su bajo consumo, combinado con técnicas de detección inteligente de eventos y predicción adaptativa de la respuesta del sistema, permite reducir drásticamente el número de transmisiones innecesarias y el tiempo de actividad de los actuadores. Estos avances suponen un claro paso hacia una energía más sostenible, facilitando que infraestructuras como plantas solares (el cual es el campo de aplicación planteado como central en el trabajo) operen de forma más eficiente y responsable.

ODS 9: INDUSTRIA, INNOVACIÓN E INFRAESTRUCTURA

El Objetivo 9 promueve la construcción de infraestructuras que sean adaptables, así como la industrialización sostenible y el fomento de la innovación. El sistema propuesto aborda directamente estos aspectos al introducir inteligencia artificial en redes de sensores a gran escala, permitiendo no solo la supervisión remota, sino también la toma de decisiones automatizada en tiempo real. Al reducir el tráfico de red y optimizar las respuestas que buscan corregir la correcta posición de los sensores, se potencia la capacidad de adaptación de las instalaciones industriales ante eventos imprevistos y se impulsa el desarrollo de infraestructuras más inteligentes, capaces de anticipar fallos y responder de manera eficiente a las variaciones en el entorno.

Además, la utilización de arquitecturas accesibles y de bajo coste aboga también por la igualdad de cara al acceso de recursos tecnológicos, facilitando que estas soluciones puedan ser adoptadas también por pequeñas y medianas empresas que buscan mejorar sus procesos productivos de la manera más sencilla posible.

ODS 11: CIUDADES Y COMUNIDADES SOSTENIBLES

Aunque el proyecto está enfocado inicialmente a aplicaciones industriales y energéticas, su arquitectura modular y escalable lo hace exportable a entornos urbanos. Redes de sensores optimizadas como las desarrolladas pueden aplicarse en ciudades inteligentes para monitorizar infraestructuras de todo tipo, gestionar de forma más eficiente los recursos energéticos de estas o detectar anomalías estructurales en edificios. Así, se contribuye al Objetivo 11, fomentando comunidades más seguras y sostenibles.

ODS 12: PRODUCCIÓN Y CONSUMO RESPONSABLES

El planteamiento del proyecto persigue, entre otros objetivos, el uso responsable de los recursos tecnológicos y energéticos. Reducir el volumen de datos transmitidos y optimizar el número de acciones ejecutadas por los actuadores disminuye significativamente el desgaste de los dispositivos, alarga su vida útil y reduce las necesidades de mantenimiento y sustitución. Estas mejoras, alineadas con el Objetivo 12, promueven un ciclo de vida más sostenible de los sistemas electrónicos y una producción más eficiente y respetuosa con el medio ambiente.

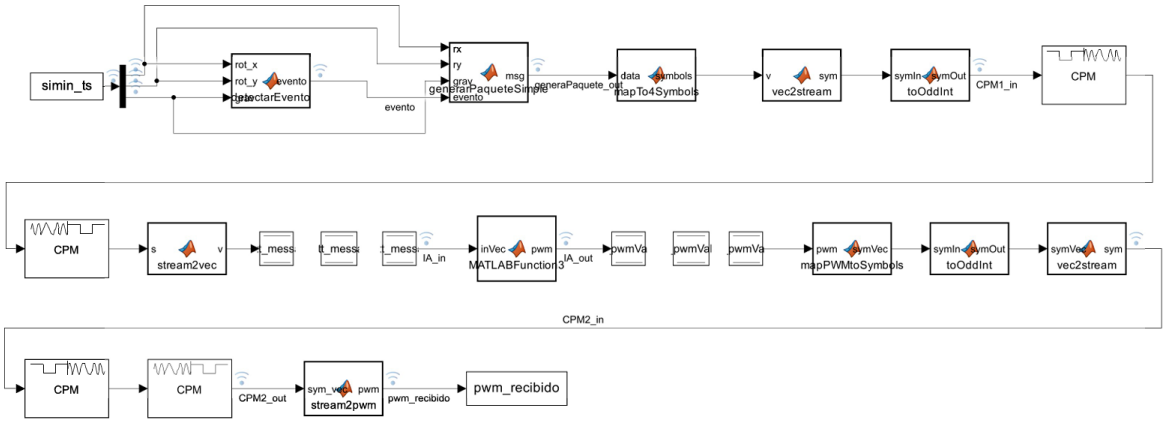
ODS 13: ACCIÓN POR EL CLIMA

Finalmente, al contribuir a la optimización de instalaciones de energía renovable, como las plantas solares, el proyecto incide de forma directa en el Objetivo 13 de lucha contra el cambio climático. Incrementar la eficiencia de estos sistemas permite maximizar la producción energética limpia y minimizar las emisiones indirectas asociadas al mantenimiento y operación de estas infraestructuras. Además, la capacidad predictiva del sistema facilita una gestión más proactiva ante fenómenos extremos o variaciones ambientales, aspectos cada vez más críticos en un escenario de cambio climático acelerado.

En conclusión, la solución presentada en este trabajo de fin de grado no solo aborda un reto tecnológico, sino que ofrece respuestas concretas a desafíos globales recogidos en los ODS. A través de la combinación de tecnologías IoT, protocolos de comunicación eficientes y modelos de inteligencia artificial, se propone un sistema que contribuye de forma activa al avance hacia una industria más sostenible, una gestión energética más responsable y a un desarrollo tecnológico que busca no comprometer el futuro.

ANEXO II

[1] Flujo complete de Simulink



[2] Función IA_predict de Matlab para llamar a la predicción hecha en Python

```
function PWM = IA_predict(inVec)
%#codegen
 arguments
  inVec (1,4) double
 end

 coder.extrinsic('py.predict_PWM.predict_PWM');

 PWM = 0.0;

 rot_x  = inVec(1);
 rot_y  = inVec(2);
 grav   = inVec(3);
 evento = inVec(4);

 if evento == 1.0
  PWM_py = py.predict_PWM.predict_PWM(rot_x, rot_y, grav, int32(1));
  PWM    = double(PWM_py);
 end

 PWM = min(max(PWM, 0), 120);
end
```

[3] Función predict_PWM.py en Python para cargar el algoritmo de IA en Simulink

```
import os
import joblib
import numpy as np
from collections import deque

BASE_DIR = os.path.dirname(__file__)
scaler, model, FEATURES = joblib.load(
    os.path.join(BASE_DIR, "PWM_regressor_events_static.pkl")
)

KEEPALIVE = 0.0

def predict_PWM(rot_x: float, rot_y: float, grav: float, evento: int) -> float:
    if evento == 0:
        return 0.0

    vals = {
        'rot_x': rot_x,
        'rot_y': rot_y,
        'grav': grav,
        'rot_x_rot_y': rot_x * rot_y,
        'rot_x_grav':  rot_x * grav,
        'rot_y_grav':  rot_y * grav,
    }

    try:
        X = np.array([[vals[f] for f in FEATURES]], dtype=float)
        Xs = scaler.transform(X)
        PWM = model.predict(Xs)[0]
        return float(np.clip(PWM, 0.0, 120.0))
    except Exception as e:
        print("Error IA:", e)
        return 0.0
```

[4] Código para la detección de eventos en Python

```
import pandas as pd
import numpy as np
from sklearn.ensemble import IsolationForest
from skimage.filters import threshold_otsu
import matplotlib.pyplot as plt

DATA_CSV  = "datos_Sensores.csv" # Datos previamente limpios y verificados
OUT_CSV   = "datos_Sensores_con_evento_def.csv"
SEED      = 42
```

```
TARGET_PCT = None  # Cuando pone None utilizamos Otsu

# Cargar datos
df = pd.read_csv(DATA_CSV, sep=";", decimal=",")
print(f"Cargadas {len(df):,} filas")

# Ingeniería de características
for col in ['rot_x', 'rot_y', 'grav']:
    df[f'{col}_diff'] = df[col].diff().fillna(0)
df['rot_x_rot_y'] = df['rot_x'] * df['rot_y']
df['rot_x_grav']  = df['rot_x'] * df['grav']
df['rot_y_grav']  = df['rot_y'] * df['grav']

# Detección de los eventos con Isolation Forest
detector_feat = ['rot_x', 'rot_y', 'grav', 'PWMValue1']
iso = IsolationForest(n_estimators=200, contamination='auto', random_state=SEED)
iso.fit(df[detector_feat])

score = -iso.decision_function(df[detector_feat])  # Cuanto mayor, más raro
score = (score - score.min()) / (score.max() - score.min() + 1e-12)  # Normalizado

# Umbral de decisión adaptable
if TARGET_PCT is None:
    thr = threshold_otsu(score)
    print(f"Umbral Otsu = {thr:.4f}")
else:
    thr = np.percentile(score, 100 * (1 - TARGET_PCT))
    print(f"Umbral manual = {thr:.4f}")

df['evento'] = (score >= thr).astype(int)
print(f"Porcentaje de eventos detectados: {df['evento'].mean():.2%}")

# Guardar CSV
df.to_csv(OUT_CSV, sep=';', decimal=',', index=False)
print(f"CSV con eventos guardado en: {OUT_CSV}")

# Análisis estadístico
stats = df.groupby('evento')[['rot_x', 'rot_y', 'grav', 'PWMValue1']].describe()
print("\nResumen estadístico:")
print(stats.loc[:, (slice(None), ['mean', 'std', 'min', 'max'])])

# Percentiles calculados
percentiles = df.groupby('evento')[['rot_x', 'rot_y', 'grav', 'PWMValue1']].quantile([0.05, 0.25, 0.5,
0.75, 0.95])
print("\nPercentiles clave por clasificación de evento:")
print(percentiles)
```

[5] Código para el modelo de predicción de PWM en Python

```
import pandas as pd
import numpy as np
from sklearn.ensemble import HistGradientBoostingRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_absolute_error, r2_score
import joblib

DATA_CSV  = "datos_Sensores_con_evento_def.csv"
MODELO_PKL = "PWM_regressor_events_balanced.pkl"

# Cargar los datos con los eventos
df = pd.read_csv(DATA_CSV, sep=';', decimal=',')

# Hacer un split temporal antes de balancear (para que no haya sobreajuste por tramos)
split = int(len(df_evt) * 0.7)
df_train = df_evt.iloc[:split].copy()
df_test  = df_evt.iloc[split:].copy()

# Balancear solo el set de entrenamiento
pico_mask = (df_train['PWMValue1'] >= 64) & (df_train['PWMValue1'] <= 66)
otros_mask = ~pico_mask

df_pico = df_train[pico_mask].sample(frac=0.1, random_state=42)
df_train_balanced = pd.concat([df_train[otros_mask], df_pico]).sample(frac=1,
random_state=SEED)
print(f"Dataset de entrenamiento balanceado: {len(df_train_balanced):,}")
print(f"Dataset de test real sin balancear: {len(df_test):,}")

reg_feat = [
    'rot_x', 'rot_y', 'grav',
    'rot_x_diff', 'rot_y_diff', 'grav_diff',
    'rot_x_rot_y', 'rot_x_grav', 'rot_y_grav'
]

X_tr = df_train_balanced[reg_feat].values
y_tr = df_train_balanced['PWMValue1'].values

X_te = df_test[reg_feat].values
y_te = df_test['PWMValue1'].values

# Entrenar el modelo
pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('reg', HistGradientBoostingRegressor(
        max_iter=300,
        learning_rate=0.05,
```

```
        max_depth=10,
        l2_regularization=0.1,
        random_state=SEED
    ))
])

pipeline.fit(X_tr, y_tr)
y_pred = pipeline.predict(X_te)

# Evaluar el modelo sobre el test real (sin balancear)
print("\nMétricas de PWM (test real)")
print(f"MAE  : {mean_absolute_error(y_te, y_pred):.2f}")
print(f"R²   : {r2_score(y_te, y_pred):.3f}")
print(f"Ntest: {len(y_te):,}")

# Guardar el modelo
joblib.dump((pipeline.named_steps['scaler'], pipeline.named_steps['reg'], reg_feat),
MODEL_PKL)
df_train_balanced.to_csv(OUT_CSV, sep=';', decimal=',', index=False)
print(f"\nModelo guardado en → {MODEL_PKL}")
```