

Universidad de Costa Rica

Laboratorio de Microcontroladores

Laboratorio #5:  
STM32/Arduino: GPIO, Giroscopio, comunicaciones,  
TinyML

Prof. MSc. Marco Villalta Fallas

Marco Vásquez Ovares - B17032  
Junior Ruiz Sánchez - B97026

Grupo: 01

I Ciclo 2025

# Índice

<b>1. Introducción</b>	<b>5</b>
<b>2. Nota teórica</b>	<b>6</b>
2.1. Arduino . . . . .	6
2.2. Arduino Nano 33 BLE Sense Lite . . . . .	6
2.2.1. Diagrama de pines . . . . .	8
2.2.2. Descripción de pines . . . . .	9
2.2.3. Características eléctricas . . . . .	9
2.3. Microcontrolador nRF52840 . . . . .	11
2.3.1. Diagrama de pines . . . . .	12
2.3.2. Descripción de pines . . . . .	13
2.3.3. Diagrama de bloques . . . . .	16
2.3.4. Características eléctricas . . . . .	17
2.4. TensorFlow Lite . . . . .	18
2.5. Edge Impulse . . . . .	19
2.6. Human Activity Recognition (HAR) . . . . .	20
2.7. Machine Learning (ML) . . . . .	21
2.8. Diseño del circuito . . . . .	22
2.8.1. Lista de componentes . . . . .	22
2.8.2. Información adicional . . . . .	22
<b>3. Desarrollo</b>	<b>24</b>
3.1. Flujo de captura de datos . . . . .	24
3.2. Flujo del firmware de clasificación . . . . .	25
3.3. Análisis del firmware . . . . .	25
3.3.1. Firmware . . . . .	25
3.4. Análisis electrónico . . . . .	26
3.5. Análisis de resultados . . . . .	27
3.5.1. Movimiento: Círculos . . . . .	27
3.5.2. Movimiento: Arriba-Abajo . . . . .	28
3.5.3. Movimiento: Golpe . . . . .	29
3.5.4. Gráfica de la pérdida . . . . .	29
3.5.5. Gráfico de la pérdida nuevamente, saltando un poco de inicio . . . . .	30
3.5.6. Gráfica el error absoluto medio . . . . .	30
3.5.7. Trazar las predicciones junto con los datos de la prueba . . . . .	31
3.6. Edge Impulse . . . . .	32
3.6.1. Data acquisition . . . . .	32
3.6.2. Create Impulse . . . . .	32
3.6.3. Raw data . . . . .	33
3.6.4. Classifier . . . . .	34
3.6.5. Model testing . . . . .	35
3.6.6. Deployment . . . . .	37
3.6.7. Resultados . . . . .	38

<b>4. Conclusiones y Recomendaciones</b>	<b>39</b>
4.1. Conclusiones . . . . .	39
4.2. Recomendaciones . . . . .	39
<b>Bibliografía</b>	<b>40</b>
<b>5. Apéndices</b>	<b>40</b>
5.1. Repositorio Git . . . . .	40

## Índice de figuras

1.	Arduino® Nano 33 BLE Sense . . . . .	6
2.	Connector Pinouts . . . . .	8
3.	Description . . . . .	9
4.	Power Consumption . . . . .	9
5.	Power Tree . . . . .	10
6.	Diagrama de pines . . . . .	12
7.	Descripción de pines . . . . .	13
8.	Descripción de pines . . . . .	14
9.	Descripción de pines . . . . .	15
10.	Diagrama de bloques . . . . .	16
11.	Absolute maximum ratings . . . . .	17
12.	kit Arduino Nano 33 BLE Sense Lite . . . . .	22
13.	Circuito final . . . . .	26
14.	aceleración-círculos . . . . .	27
15.	giroscopio-círculos . . . . .	27
16.	aceleración-Arriba_Aabajo . . . . .	28
17.	giroscopio-Arriba_Aabajo . . . . .	28
18.	aceleración-golpe . . . . .	29
19.	giroscopio-golpe . . . . .	29
20.	Graph the loss . . . . .	30
21.	Graph the loss again, skipping a bit of the start . . . . .	30
22.	Graph the mean absolute error . . . . .	31
23.	Plot the predictions along with to the test data . . . . .	31
24.	data . . . . .	32
25.	Impulse . . . . .	33
26.	Features . . . . .	33
27.	Features explorer . . . . .	34
28.	Last training performance . . . . .	35
29.	Data explorer . . . . .	35
30.	Results . . . . .	36
31.	Results . . . . .	36
32.	build . . . . .	37
33.	resultados-serial monitor . . . . .	38
34.	resultados-serial monitor . . . . .	38

## Índice de tablas

1.	Lista de componentes . . . . .	22
----	--------------------------------	----

# 1. Introducción

## Resumen

El documento describe el desarrollo de un sistema de **Reconocimiento de Actividad Humana (HAR)** sobre la **Arduino Nano 33 BLE Sense Lite**. El objetivo consiste en que el microcontrolador **nRF52840** identifique tres gestos de brazo —círculos, arriba/abajo y golpe— mediante un modelo de *Tiny Machine Learning* ejecutado localmente. El código se compila y se carga en la placa desde el **Arduino IDE**.

El informe aborda, de forma introductoria y aplicada, los temas siguientes:

- Funcionamiento de la **IMU LSM9DS1** integrada y su uso para capturar aceleraciones y giros.
- Principios de **TinyML** y la variante **TensorFlow Lite for Microcontrollers**, incluidas técnicas de cuantización para reducir recursos.
- Utilización de **Edge Impulse** para la adquisición de datos, el entrenamiento del modelo y la generación de la librería compatible con Arduino.
- Flujo estándar de un proyecto **HAR**: captura de datos, preprocesamiento, extracción de características y clasificación.
- Montaje mínimo del hardware (placa y cable USB) y validación del modelo en tiempo real y de forma local.

## 2. Nota teórica

### 2.1. Arduino

**Arduino** es una plataforma de desarrollo de hardware libre basada en una familia de placas electrónicas con microcontrolador, acompañada de un entorno de programación sencillo y accesible. Su principal objetivo es facilitar la creación de prototipos electrónicos interactivos, permitiendo que estudiantes, ingenieros, artistas y entusiastas de la tecnología puedan desarrollar sistemas embebidos sin necesidad de conocimientos profundos en electrónica o programación a bajo nivel.

Una de las características más importantes de Arduino es su entorno de desarrollo integrado (IDE), el cual utiliza un lenguaje basado en C/C++ simplificado, junto con una gran variedad de bibliotecas que permiten controlar sensores, actuadores y periféricos de manera directa y eficiente. Además, Arduino cuenta con una comunidad activa que proporciona documentación, ejemplos y soporte.

Las placas Arduino más comunes incluyen microcontroladores como el ATmega328P (Arduino Uno), ATmega2560 (Mega), o procesadores más avanzados como el nRF52840 en el caso del Arduino Nano 33 BLE, que además integra sensores como acelerómetro, giroscopio y conectividad Bluetooth.

Arduino se ha consolidado como una herramienta clave en el ámbito educativo, la investigación y el desarrollo de soluciones de bajo costo, debido a su enfoque práctico, modular y accesible. Gracias a su naturaleza de código abierto, es posible tanto modificar el hardware como adaptar el software a proyectos específicos, lo que ha impulsado su adopción en todo tipo de aplicaciones, desde sistemas de monitoreo ambiental y automatización del hogar hasta desarrollos con inteligencia artificial embebida.

### 2.2. Arduino Nano 33 BLE Sense Lite

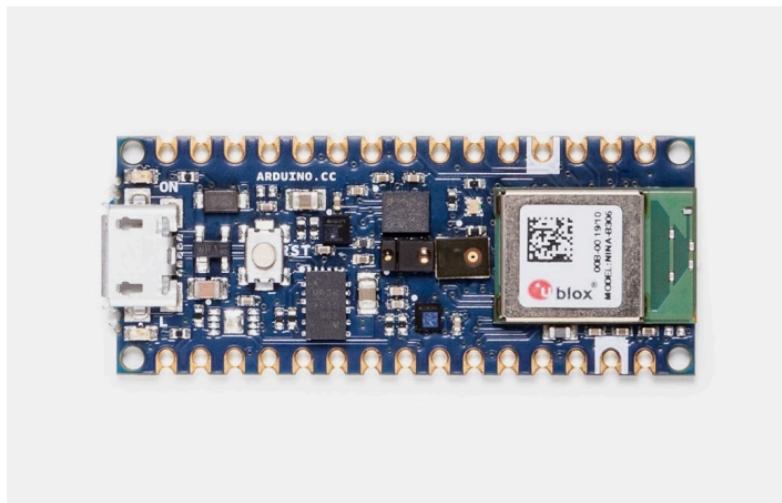


Figura 1: Arduino® Nano 33 BLE Sense

El **Arduino Nano 33 BLE Sense Lite** es una placa de desarrollo compacta y poderosa perteneciente a la familia Arduino Nano, diseñada para aplicaciones modernas que requieren conectividad inalámbrica e inteligencia artificial embebida. Está basada en el microcontrolador nRF52840 de Nor-

dic Semiconductor, que cuenta con un procesador ARM Cortex-M4 de 32 bits con punto flotante, funcionando a 64 MHz.

Una de sus principales ventajas es la integración de sensores avanzados directamente en la placa, lo que permite capturar información del entorno sin hardware adicional. Incluye un sensor IMU de 9 ejes (acelerómetro, giroscopio y magnetómetro), un micrófono digital, y otros sensores opcionales en la versión estándar (como temperatura, humedad y presión). En la versión "Lite", la funcionalidad es similar pero con un enfoque más liviano, manteniendo el núcleo BLE y el IMU.

La conectividad Bluetooth Low Energy (BLE) permite que el dispositivo se comunique con otros dispositivos móviles o sensores sin necesidad de cables, siendo ideal para aplicaciones como monitoreo biométrico, reconocimiento de gestos, wearables y sistemas de interacción con el entorno.

Esta placa es compatible con TensorFlow Lite for Microcontrollers, lo que permite ejecutar modelos de aprendizaje automático directamente en el microcontrolador. De esta manera, es posible implementar sistemas inteligentes sin depender de la nube, ahorrando recursos y mejorando la respuesta en tiempo real.

Su diseño compacto, eficiencia energética, conectividad BLE y capacidad de ejecutar inferencias ML en el borde, hacen del Arduino Nano 33 BLE Sense Lite una opción ideal para proyectos modernos de IoT, salud, seguridad y automatización.

### **Características generales – Arduino Nano 33 BLE Sense Lite:**

- Microcontrolador: nRF52840 (ARM Cortex-M4 @ 64 MHz)
- Memoria Flash: 1 MB
- RAM: 256 KB
- Voltaje de operación: 3.3 V
- Conectividad: Bluetooth Low Energy (BLE) 5.0
- IMU: LSM9DS1 (acelerómetro, giroscopio y magnetómetro de 9 ejes)
- Puertos de comunicación: UART, SPI, I2C
- Puertos de E/S(I/O) digitales: 14
- Entradas analógicas: 8 (12 bits)
- Conector USB: Micro-USB (para programación y comunicación serial)
- Dimensiones: 45 mm x 18 mm
- Peso: 5 g

### 2.2.1. Diagrama de pines

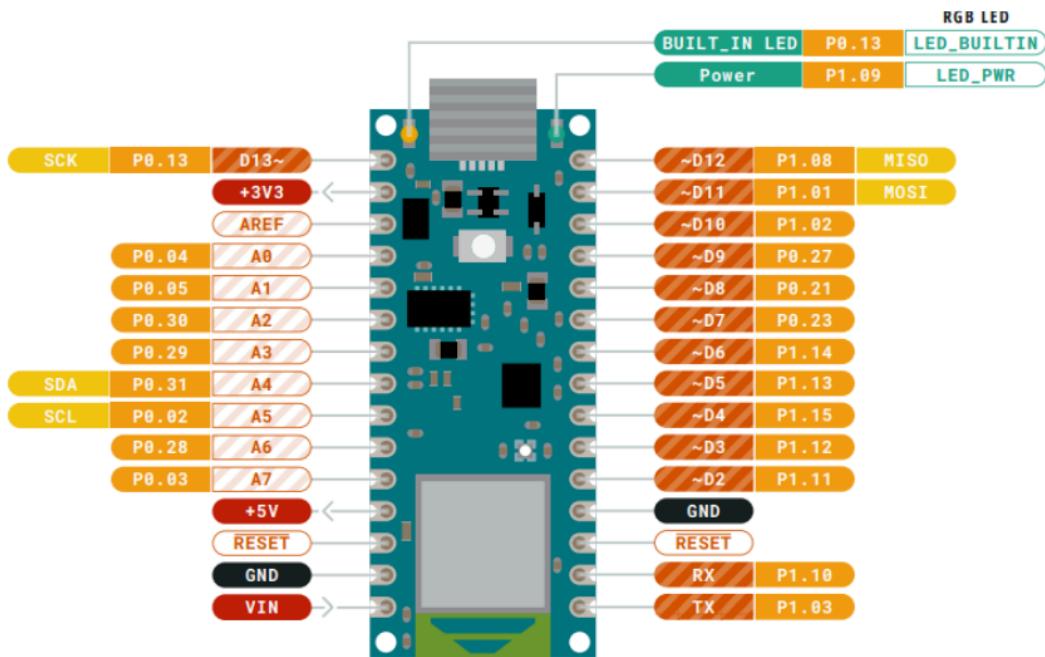


Figura 2: Connector Pinouts

## 2.2.2. Descripción de pines

<b>Pin</b>	<b>Function</b>	<b>Type</b>	<b>Description</b>
1	D13	Digital	GPIO
2	+3V3	Power Out	Internally generated power output to external devices
3	AREF	Analog	Analog Reference; can be used as GPIO
4	A0/DAC0	Analog	ADC in/DAC out; can be used as GPIO
5	A1	Analog	ADC in; can be used as GPIO
6	A2	Analog	ADC in; can be used as GPIO
7	A3	Analog	ADC in; can be used as GPIO
8	A4/SDA	Analog	ADC in; I2C SDA; Can be used as GPIO (1)
9	A5/SCL	Analog	ADC in; I2C SCL; Can be used as GPIO (1)
10	A6	Analog	ADC in; can be used as GPIO
11	A7	Analog	ADC in; can be used as GPIO
12	VUSB	Power In/Out	Normally NC; can be connected to VUSB pin of the USB connector by shorting a jumper
13	RST	Digital In	Active low reset input (duplicate of pin 18)
14	GND	Power	Power Ground
15	VIN	Power In	Vin Power input
16	TX	Digital	USART TX; can be used as GPIO
17	RX	Digital	USART RX; can be used as GPIO
18	RST	Digital	Active low reset input (duplicate of pin 13)
19	GND	Power	Power Ground
20	D2	Digital	GPIO
21	D3/PWM	Digital	GPIO; can be used as PWM
22	D4	Digital	GPIO
23	D5/PWM	Digital	GPIO; can be used as PWM
24	D6/PWM	Digital	GPIO, can be used as PWM
25	D7	Digital	GPIO
26	D8	Digital	GPIO
27	D9/PWM	Digital	GPIO; can be used as PWM
28	D10/PWM	Digital	GPIO; can be used as PWM
29	D11/MOSI	Digital	SPI MOSI; can be used as GPIO
30	D12/MISO	Digital	SPI MISO; can be used as GPIO

Figura 3: Description

## 2.2.3. Características eléctricas

### 1.2 Power Consumption

<b>Symbol</b>	<b>Description</b>	<b>Min</b>	<b>Typ</b>	<b>Max</b>	<b>Unit</b>
PBL	Power consumption with busy loop		TBC		mW
PLP	Power consumption in low power mode		TBC		mW
PMAX	Maximum Power Consumption		TBC		mW

Figura 4: Power Consumption

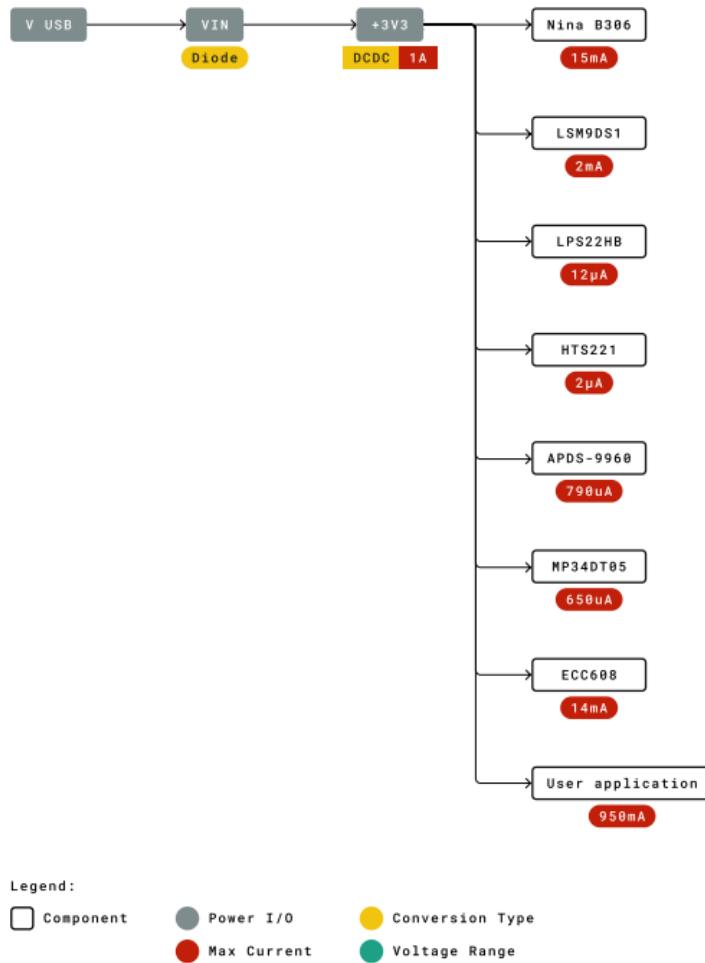


Figura 5: Power Tree

## 2.3. Microcontrolador nRF52840

El nRF52840 es un microcontrolador de alto rendimiento desarrollado por Nordic Semiconductor, diseñado específicamente para aplicaciones de conectividad inalámbrica de bajo consumo energético. Este microcontrolador forma parte de la familia nRF52 Series, ampliamente utilizada en dispositivos IoT, wearables, automatización, sensores inteligentes y sistemas embebidos con capacidades inalámbricas avanzadas.

El nRF52840 está basado en un procesador ARM Cortex-M4F de 32 bits, que incluye una unidad de punto flotante (FPU) para mejorar el rendimiento en cálculos matemáticos, especialmente útiles en procesamiento de señales, control de sensores, y ejecución de modelos de inteligencia artificial embebida (como los de TensorFlow Lite for Microcontrollers).

Una de sus características más destacadas es su soporte para múltiples protocolos de comunicación inalámbrica, incluyendo Bluetooth Low Energy (BLE) 5.0, Bluetooth Mesh, Thread, Zigbee, y ANT, así como capacidades de comunicación 2.4 GHz propietarias. Esto lo convierte en una solución extremadamente flexible para aplicaciones que requieren conectividad segura y robusta.

El nRF52840 cuenta con una arquitectura eficiente en energía y modos de bajo consumo, lo cual es ideal para dispositivos que deben funcionar durante largos períodos con baterías pequeñas. También ofrece una variedad de interfaces periféricas, incluyendo GPIO, ADC, UART, SPI, I2C, PWM y USB 2.0 de alta velocidad.

En el caso del Arduino Nano 33 BLE Sense Lite, este microcontrolador permite no solo la adquisición y procesamiento de datos en tiempo real, sino también la ejecución local de inferencias de modelos de Machine Learning, eliminando la necesidad de conectividad a la nube para tareas de reconocimiento o clasificación.

### Características generales del nRF52840:

- CPU: ARM Cortex-M4F de 32 bits @ 64 MHz (con FPU)
- Memoria Flash: 1 MB
- RAM: 256 KB
- Soporte inalámbrico:
  - Bluetooth 5.0 / BLE / Bluetooth Mesh
  - Thread, Zigbee, ANT, 2.4 GHz propietario
- Conectividad por hardware:
  - USB 2.0 (full-speed)
  - UART, SPI, I2C, I2S, PWM, QSPI, PDM
- Rango de voltaje operativo: 1.7 V – 3.6 V
- Modos de bajo consumo: consumo ultra bajo en modo sleep
- Criptografía: Acelerador hardware para AES, SHA-2, ECC
- Otros: Soporte para actualización de firmware Over-The-Air (OTA)

### 2.3.1. Diagrama de pines

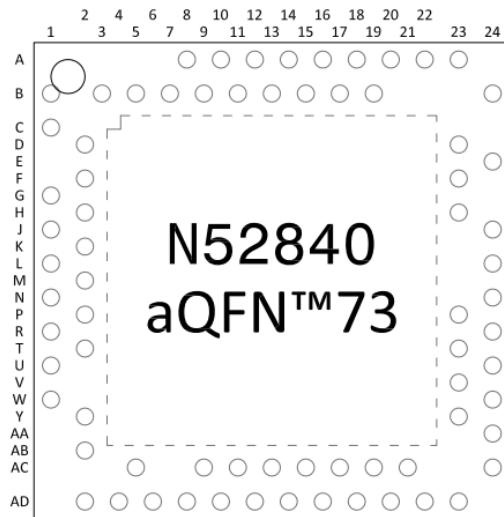


Figura 6: Diagrama de pines

### 2.3.2. Descripción de pines

Pin	Name	Function	Description	Recommended usage
A8	P0.31	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
	AIN7	Analog input	Analog input	
A10	P0.29	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
	AIN5	Analog input	Analog input	
A12	P0.02	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
	AIN0	Analog input	Analog input	
A14	P1.15	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
A16	P1.13	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
A18	DEC2	Power	1.3 V regulator supply decoupling	
A20	P1.10	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
A22	VDD	Power	Power supply	
A23	XC2	Analog input	Connection for 32 MHz crystal	
B1	VDD	Power	Power supply	
B3	DCC	Power	DC/DC converter output	
B5	DEC4	Power	1.3 V regulator supply decoupling	Must be connected to DEC6 (pin E24)
B7	VSS	Power	Ground	
B9	P0.30	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
	AIN6	Analog input	Analog input	
B11	P0.28	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
	AIN4	Analog input	Analog input	
B13	P0.03	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
	AIN1	Analog input	Analog input	
B15	P1.14	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
B17	P1.12	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
B19	P1.11	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
B24	XC1	Analog input	Connection for 32 MHz crystal	
C1	DEC1	Power	1.1 V regulator supply decoupling	
D2	P0.00	Digital I/O	General purpose I/O	
	XL1	Analog input	Connection for 32.768 kHz crystal	
D23	DEC3	Power	Power supply, decoupling	
E24	DEC6	Power	1.3 V regulator supply decoupling	Must be connected to DEC4 (pin B5)
F2	P0.01	Digital I/O	General purpose I/O	
	XL2	Analog input	Connection for 32.768 kHz crystal	
F23	VSS_PA	Power	Ground (radio supply)	
G1	P0.26	Digital I/O	General purpose I/O	
H2	P0.27	Digital I/O	General purpose I/O	
H23	ANT	RF	Single-ended radio antenna connection	See <a href="#">Reference circuitry</a> on page 937 for guidelines on how to ensure good RF performance
J1	P0.04	Digital I/O	General purpose I/O	
	AIN2	Analog input	Analog input	
J24	P0.10	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only

Figura 7: Descripción de pines

Pin	Name	Function	Description	Recommended usage
	NFC2	NFC input	NFC antenna connection	
K2	P0.05	Digital I/O	General purpose I/O	
	AIN3	Analog input	Analog input	
L1	P0.06	Digital I/O	General purpose I/O	
L24	P0.09	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
	NFC1	NFC input	NFC antenna connection	
M2	P0.07	Digital I/O	General purpose I/O	
	TRACECLK	Trace clock	Trace buffer clock	
N1	P0.08	Digital I/O	General purpose I/O	
N24	DECS	Power	1.3 V regulator supply decoupling for build codes Dxx and earlier.	
		Not connected	Not connected for build codes Fxx and later.	
P2	P1.08	Digital I/O	General purpose I/O	
P23	P1.07	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
R1	P1.09	Digital I/O	General purpose I/O	
	TRACEDATA3	Trace data	Trace buffer TRACEDATA[3]	
R24	P1.06	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
T2	P0.11	Digital I/O	General purpose I/O	
	TRACEDATA2	Trace data	Trace buffer TRACEDATA[2]	
T23	P1.05	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
U1	P0.12	Digital I/O	General purpose I/O	
	TRACEDATA1	Trace data	Trace buffer TRACEDATA[1]	
U24	P1.04	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
V23	P1.03	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
W1	VDD	Power	Power supply	
W24	P1.02	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
Y2	VDDH	Power	High voltage power supply	
Y23	P1.01	Digital I/O	General purpose I/O	Standard drive, low frequency I/O only
AA24	SWDCLK	Debug	Serial wire debug clock input for debug and programming	
AB2	DCCH	Power	DC/DC converter output	
AC5	DECLUSB	Power	USB 3.3 V regulator supply decoupling	
AC9	P0.14	Digital I/O	General purpose I/O	
AC11	P0.16	Digital I/O	General purpose I/O	
AC13	P0.18	Digital I/O	General purpose I/O	QSPI/CSN
	nRESET		Configurable as pin RESET	
AC15	P0.19	Digital I/O	General purpose I/O	QSPI/SCK
AC17	P0.21	Digital I/O	General purpose I/O	QSPI
AC19	P0.23	Digital I/O	General purpose I/O	QSPI
AC21	P0.25	Digital I/O	General purpose I/O	
AC24	SWDIO	Debug	Serial wire debug I/O for debug and programming	
AD2	VBUS	Power	5 V input for USB 3.3 V regulator	
AD4	D-	USB	USB D-	
AD6	D+	USB	USB D+	

Figura 8: Descripción de pines

Pin	Name	Function	Description	Recommended usage
AD8	P0.13	Digital I/O	General purpose I/O	
AD10	P0.15	Digital I/O	General purpose I/O	
AD12	P0.17	Digital I/O	General purpose I/O	
AD14	VDD	Power	Power supply	
AD16	P0.20	Digital I/O	General purpose I/O	
AD18	P0.22	Digital I/O	General purpose I/O	QSPI
AD20	P0.24	Digital I/O	General purpose I/O	
AD22	P1.00	Digital I/O	General purpose I/O	QSPI
	TRACEDATA0	Trace data	Trace buffer TRACEDATA[0] Serial wire output (SWO)	
AD23	VDD	Power	Power supply	
Die pad	VSS	Power	Ground pad	Exposed die pad must be connected to ground (VSS) for proper device operation

Figura 9: Descripción de pines

### 2.3.3. Diagrama de bloques

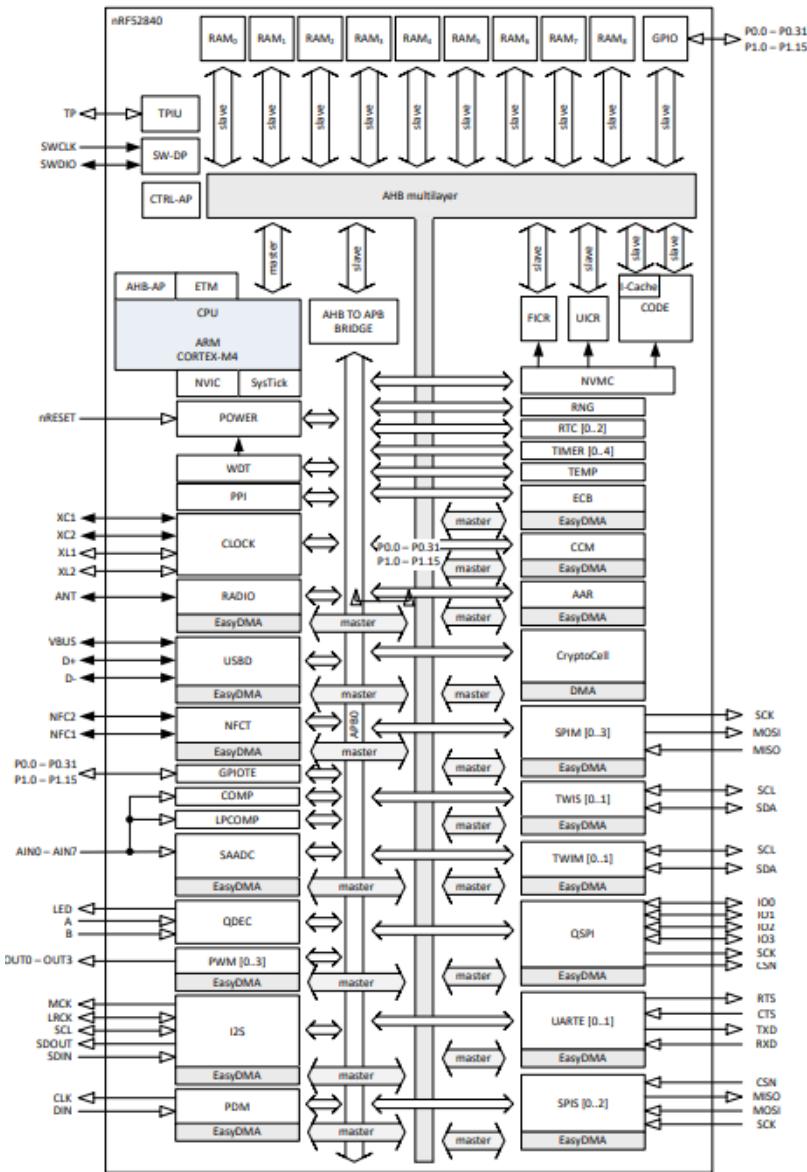


Figura 10: Diagrama de bloques

### 2.3.4. Características eléctricas

Note	Min.	Max.	Unit
<b>Supply voltages</b>			
VDD	-0.3	+3.9	V
VDDH	-0.3	+5.8	V
VBUS	-0.3	+5.8	V
VSS	0		V
<b>I/O pin voltage</b>			
V <sub>I/O</sub> , VDD ≤ 3.6 V	-0.3	VDD + 0.3	V
V <sub>I/O</sub> , VDD > 3.6 V	-0.3	3.9	V
<b>NFC antenna pin current</b>			
I <sub>NFC1/2</sub>	80		mA
<b>Radio</b>			
RF input level	10		dBm
<b>Environmental aQFN73 package</b>			
Storage temperature	-40	+125	°C
MSL	Moisture Sensitivity Level	2	
ESD HBM	Human Body Model	2	kV
ESD HBM Class	Human Body Model Class	2	
ESD CDM	Charged Device Model	450	V
<b>Environmental QFN48 package</b>			
Storage temperature	-40	+125	°C
MSL	Moisture Sensitivity Level	2	
ESD HBM	Human Body Model	4	kV
ESD HBM Class	Human Body Model Class	3A	
ESD CDM	Charged Device Model	1	kV
<b>Environmental WLCSP 3.544 x 3.607 mm package</b>			
Storage temperature	-40	+125	°C
MSL	Moisture Sensitivity Level	1	
ESD HBM	Human Body Model	1	kV
ESD HBM Class	Human Body Model Class	1C	
ESD CDM	Charged Device Model	500	V
<b>Flash memory</b>			
Endurance	10 000		write/erase cycles
Retention at 85 °C	10		years

Figura 11: Absolute maximum ratings

## 2.4. TensorFlow Lite

**TensorFlow Lite (TFLite)** es una versión optimizada del framework de aprendizaje automático **TensorFlow**, diseñada específicamente para dispositivos con recursos limitados como microcontroladores, teléfonos móviles, y otros sistemas embebidos. Su objetivo es permitir la ejecución eficiente y en tiempo real de modelos de inteligencia artificial (IA) directamente en el dispositivo, sin depender de la nube o de servidores externos.

Una de las principales ventajas de TensorFlow Lite es su capacidad para ejecutar modelos previamente entrenados de forma local, lo que reduce la latencia, ahorra ancho de banda y mejora la privacidad de los datos. Esto lo convierte en una solución ideal para aplicaciones como el reconocimiento de gestos, clasificación de sonidos, detección de anomalías, visión por computadora y más, en entornos como wearables, robots, dispositivos médicos o sistemas IoT.

Para funcionar en dispositivos de muy bajo consumo, como los basados en microcontroladores (por ejemplo, el Arduino Nano 33 BLE), se utiliza **TensorFlow Lite for Microcontrollers (TFLM)**. Esta versión aún más reducida no requiere sistema operativo, ni gestión dinámica de memoria (como malloc o new), y puede integrarse directamente en firmware embebido. TFLM convierte los modelos entrenados en un formato compacto (`.tflite`) que puede almacenarse como un arreglo en memoria (`model.h`) y ejecutarse utilizando un intérprete ligero.

El flujo de trabajo general consiste en entrenar un modelo en una computadora (por ejemplo, en Python con TensorFlow), convertirlo al formato `.tflite` y luego integrarlo en el entorno de desarrollo embebido. En proyectos como este laboratorio, se puede usar incluso Edge Impulse, que automatiza la generación del modelo, su exportación como librería Arduino, y su integración con sensores como el IMU.

### Características clave de TensorFlow Lite:

- Diseñado para dispositivos con recursos limitados
- Soporte para inferencia local en tiempo real
- Compatible con modelos de clasificación, regresión, detección y más
- Versión Microcontroller (TFLM) compatible con placas como Arduino Nano 33 BLE
- No requiere sistema operativo ni asignación dinámica de memoria
- Integración sencilla con plataformas como Edge Impulse o el IDE de Arduino
- Reducción significativa en tamaño de modelos y uso de RAM/flash

## 2.5. Edge Impulse

**Edge Impulse** es una plataforma en línea especializada en el desarrollo de soluciones de inteligencia artificial (IA) para dispositivos embebidos y de bajo consumo energético. Su propósito es facilitar el diseño, entrenamiento, validación e implementación de modelos de Machine Learning (ML) directamente en microcontroladores, sensores y otros dispositivos que operan en el borde de la red ("edge devices").

A través de una interfaz web intuitiva y herramientas de línea de comandos, Edge Impulse permite a los usuarios recolectar datos desde sensores como acelerómetros, micrófonos o cámaras, etiquetarlos, y construir modelos entrenados para tareas como clasificación de gestos, reconocimiento de sonido, detección de anomalías, entre otros. Lo más destacable es que todo el proceso está orientado a ejecutarse en tiempo real directamente en el hardware, sin necesidad de conexión constante a la nube.

Edge Impulse ofrece soporte directo para plataformas como el Arduino Nano 33 BLE Sense, permitiendo capturar datos desde su sensor IMU (LSM9DS1), entrenar modelos de clasificación de movimiento, y luego exportar el modelo entrenado como una librería de Arduino (.zip) lista para compilar en el entorno del microcontrolador. Esta integración reduce significativamente la complejidad del ciclo de desarrollo en sistemas embebidos con IA.

La plataforma también proporciona herramientas para visualizar la calidad de los datos, validar el rendimiento del modelo, y optimizarlo para su uso en tiempo real. Además, Edge Impulse incluye versiones optimizadas de TensorFlow Lite for Microcontrollers (TFLM), lo que garantiza compatibilidad con dispositivos con memoria limitada.

### Características clave de Edge Impulse:

- Plataforma basada en la web para desarrollo de IA embebida
- Soporte para sensores de movimiento, audio, imagen, y más
- Herramientas integradas para captura, etiquetado y entrenamiento de datos
- Exportación directa como librería Arduino o firmware binario
- Uso de modelos optimizados con TensorFlow Lite y TFLM
- Compatible con microcontroladores como Arduino Nano 33 BLE Sense
- Visualización de precisión, rendimiento y validación del modelo
- Interfaz de línea de comandos (`edge-impulse-cli`) para desarrollo local

## 2.6. Human Activity Recognition (HAR)

**Human Activity Recognition (HAR)**, o Reconocimiento de Actividad Humana, es una rama del aprendizaje automático y la computación ubicua que se encarga de identificar y clasificar patrones de movimiento del cuerpo humano mediante el uso de sensores, algoritmos de procesamiento de señales y modelos de inteligencia artificial.

En términos prácticos, HAR permite que un sistema reconozca si una persona está caminando, corriendo, sentada, levantando el brazo, cayendo, haciendo gestos específicos, entre otras acciones, en función de los datos obtenidos de sensores como acelerómetros y giroscopios (IMU). Este tipo de reconocimiento se aplica en áreas como salud, seguridad, deportes, domótica, interfaces gestuales y monitoreo de pacientes, entre otros.

El proceso típico de HAR se basa en cuatro etapas fundamentales:

- **Adquisición de datos:** mediante sensores iniciales (IMU) integrados en dispositivos como teléfonos móviles, smartwatches o placas como el Arduino Nano 33 BLE Sense.
- **Preprocesamiento:** limpieza de datos, normalización y filtrado para eliminar ruido o inconsistencias.
- **Extracción y selección de características:** se identifican patrones en los datos que permiten distinguir entre distintas actividades.
- **Clasificación:** se utiliza un modelo de machine learning (como una red neuronal) para predecir la actividad en función de las características extraídas.

Una herramienta clave en HAR embebido es TensorFlow Lite for Microcontrollers, que permite ejecutar modelos de clasificación directamente en dispositivos con recursos limitados, como el microcontrolador nRF52840 de la placa Arduino. Además, plataformas como Edge Impulse simplifican el proceso de recolección de datos, entrenamiento y exportación de modelos para este tipo de aplicaciones.

En el contexto de este laboratorio, HAR se implementa para reconocer movimientos del brazo como 'círculos', 'arriba\_abajo' y 'golpe', utilizando datos del sensor IMU, entrenando un modelo, y ejecutándolo directamente en el microcontrolador para tomar decisiones en tiempo real sin necesidad de una conexión a la nube.

## 2.7. Machine Learning (ML)

**Machine Learning (ML)**, o Aprendizaje Automático, es una rama de la inteligencia artificial que permite a las computadoras aprender de los datos y tomar decisiones sin haber sido programadas explícitamente para cada tarea específica. En lugar de seguir reglas predefinidas, los modelos de ML identifican patrones, correlaciones y comportamientos en grandes volúmenes de datos, ajustando sus parámetros internamente para mejorar su precisión con el tiempo.

El proceso de ML se divide comúnmente en tres etapas:

- **Entrenamiento:** el modelo se alimenta con un conjunto de datos etiquetados (conocidos como *dataset de entrenamiento*) y ajusta internamente sus pesos para minimizar el error.
- **Validación:** se utiliza un subconjunto diferente de los datos para ajustar hiperparámetros y evitar sobreajuste.
- **Pruebas:** se evalúa el modelo final con datos no vistos anteriormente para medir su rendimiento real.

Existen varios tipos de algoritmos de ML, entre ellos:

- **Supervisado** (como clasificación y regresión),
- **No supervisado** (como agrupamiento o reducción de dimensiones),
- **Reforzado**, usado en sistemas autónomos.

Una de las aplicaciones más relevantes hoy en día es el uso de ML en dispositivos embebidos o de bajo consumo energético. Gracias a herramientas como TensorFlow Lite for Microcontrollers y plataformas como Edge Impulse, es posible implementar modelos livianos en microcontroladores para tareas como reconocimiento de gestos, monitoreo de salud, detección de eventos y control inteligente, todo sin conexión a Internet.

En este laboratorio, se ha aplicado ML para desarrollar un sistema de **reconocimiento de movimientos humanos (HAR)**. El modelo fue entrenado para identificar patrones de movimiento específicos del sensor IMU, como "círculo", "arriba\_abajo" y "golpe", y se integró posteriormente en el microcontrolador para su ejecución local en tiempo real.

## 2.8. Diseño del circuito

A continuación se presenta el diseño final del circuito; y en este caso el circuito corresponde únicamente al **kit Arduino Nano 33 BLE Sense Lite**.

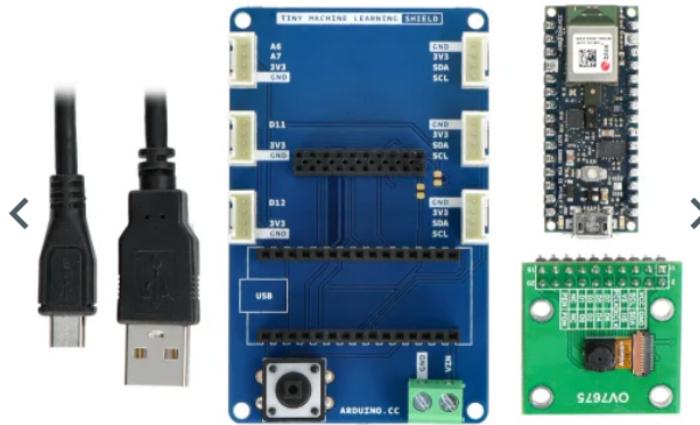


Figura 12: kit Arduino Nano 33 BLE Sense Lite  
*imagen representativa*

### 2.8.1. Lista de componentes

Tabla 1: Lista de componentes

Componente	Cantidad	Precio
kit Arduino Nano 33 BLE Sense Lite	1	US\$ 60

### 2.8.2. Información adicional

#### Red(es) Neuronal(es)

Las **redes neuronales artificiales (ANN, por sus siglas en inglés)** son un tipo de modelo de Machine Learning inspirado en la estructura del cerebro humano. Están compuestas por un conjunto de nodos interconectados llamados neuronas artificiales, organizadas en capas. Cada neurona recibe datos de entrada, los procesa aplicando una función matemática (función de activación), y transmite una salida a las neuronas de la siguiente capa.

Una red neuronal básica se estructura en tres tipos de capas:

- **Capa de entrada:** recibe los datos originales (por ejemplo, valores del acelerómetro y giroscopio).

- **Capas ocultas:** procesan la información mediante pesos y funciones de activación (como ReLU) que permiten modelar relaciones complejas.
- **Capa de salida:** entrega el resultado final, por ejemplo, la clasificación de un movimiento.

Durante el entrenamiento, la red ajusta automáticamente los pesos de las conexiones entre neuronas para minimizar el error entre la predicción y la etiqueta real, usando algoritmos como **RMSprop**, **Adam** o **SGD** y funciones de pérdida como el **MAE (Mean Absolute Error)** o **categorical cross-entropy**.

En el contexto de sistemas embebidos como el Arduino Nano 33 BLE, se emplean redes neuronales pequeñas y optimizadas para que puedan ejecutarse en microcontroladores con recursos limitados. Por ejemplo, una red de dos capas —una oculta y una de salida— puede ser suficiente para reconocer gestos a partir de datos del IMU. Estas redes son exportadas en formato TensorFlow Lite (**.tflite**) para su integración con bibliotecas como TensorFlow Lite for Microcontrollers (TFLM).

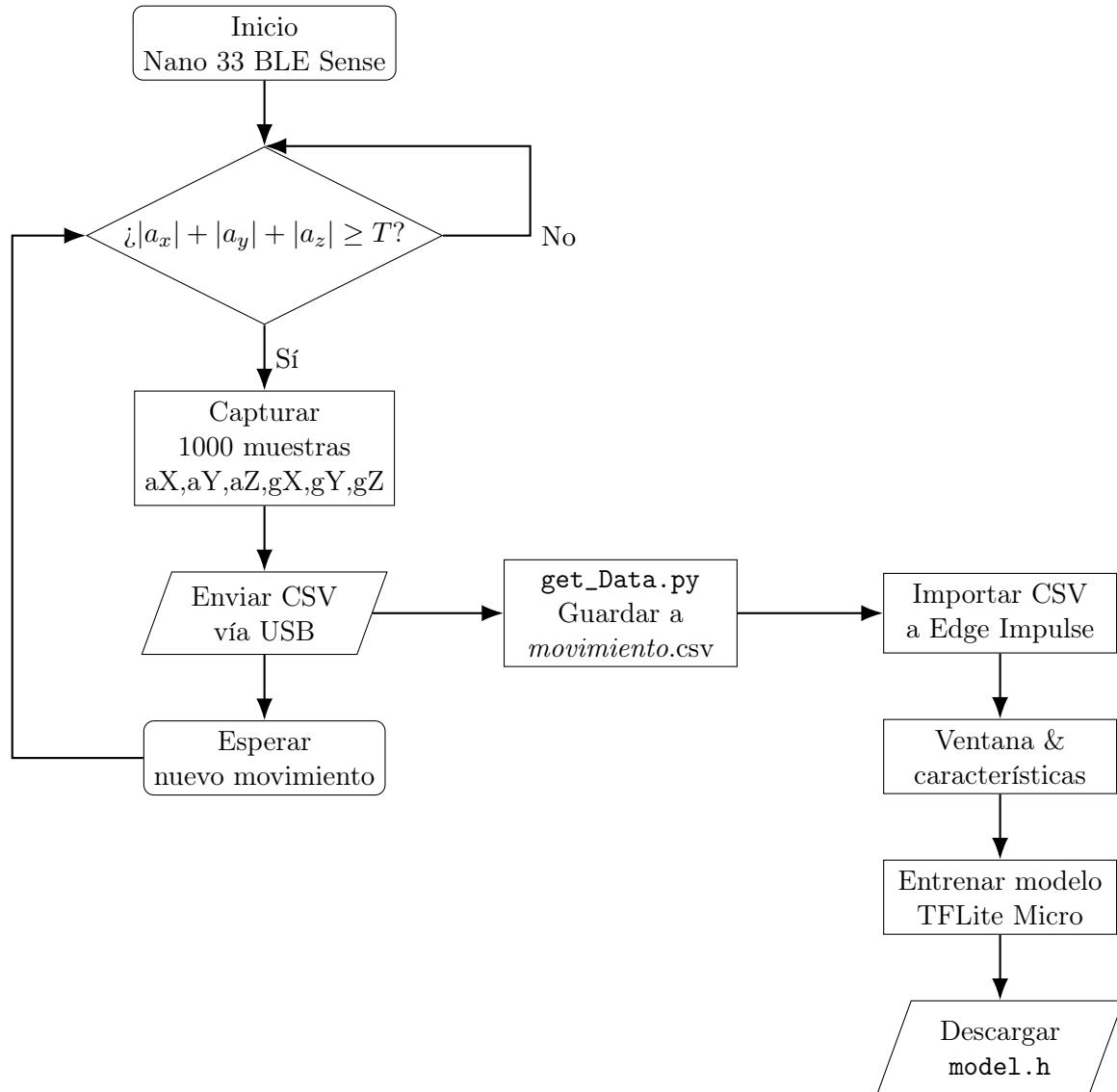
Gracias a este enfoque, es posible ejecutar inferencias de redes neuronales directamente en el dispositivo, permitiendo sistemas inteligentes sin necesidad de conectividad a la nube.

### Características clave de una red neuronal

- Entrada: datos del sensor (como aceleración o velocidad angular)
- Capas ocultas: procesan y transforman los datos con funciones como ReLU
- Salida: predicción (por ejemplo, "círculos", "arriba\_abajo", "golpe")
- Entrenamiento: ajusta pesos usando algoritmos como RMSprop
- Capacidad para aprender patrones complejos y no lineales
- En sistemas embebidos: redes pequeñas, eficientes y optimizadas

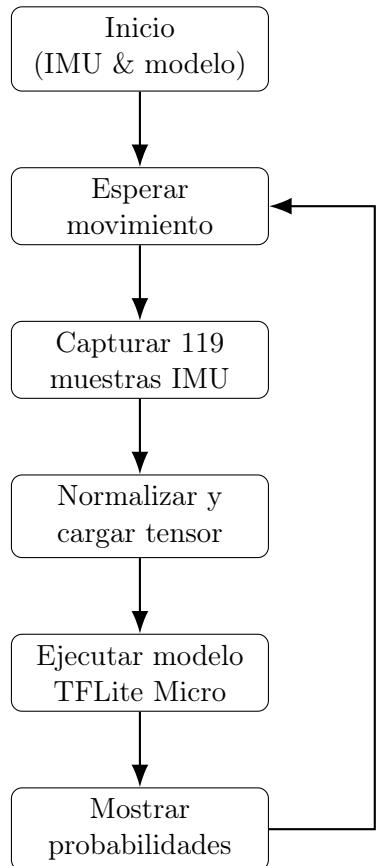
### 3. Desarrollo

#### 3.1. Flujo de captura de datos



Este diagrama muestra cómo se realizó la captura y procesamiento de datos para obtener el modelo.h. Primeramente, se obtuvieron los datos desde el arduino. Estos fueron leídos usando un código de python para crear un csv, el cual posteriormente se procesó mediante edge impulse.

### 3.2. Flujo del firmware de clasificación



Este diagrama ilustra el funcionamiento del firmware principal que ejecuta la clasificación en tiempo real. Una vez inicializados la IMU y el modelo, la placa captura los datos de los sensores de IMU. Los datos se normalizan y se cargan en el tensor de entrada. El modelo de TensorFlow Lite Micro procesa la ventana y devuelve la probabilidad de cada gesto, las cuales se imprimen por puerto serie antes de que el ciclo vuelva a la fase de espera.

### 3.3. Análisis del firmware

#### 3.3.1. Firmware

Para facilitar el proceso de prueba del modelo y su integración con el microcontrolador, se utilizó un firmware de ejemplo provisto por la plataforma Edge Impulse, específicamente adaptado para la placa Arduino Nano 33 BLE Sense. Este firmware ya incluye la estructura necesaria para capturar datos del IMU, ejecutar inferencias con el modelo entrenado en formato TensorFlow Lite, y mostrar los resultados a través del puerto serial.

El uso de este software de ejemplo permitió una implementación rápida y confiable del modelo, sirviendo como base para validar el funcionamiento del sistema de reconocimiento de movimientos en tiempo real.

### 3.4. Análisis electrónico

Como se menciona anteriormente, en cuanto al análisis electrónico, este consiste única y exclusivamente del Arduino así como un cable USB para conectarla a la computadora y realizar la comunicación entre el microcontrolador y la PC.

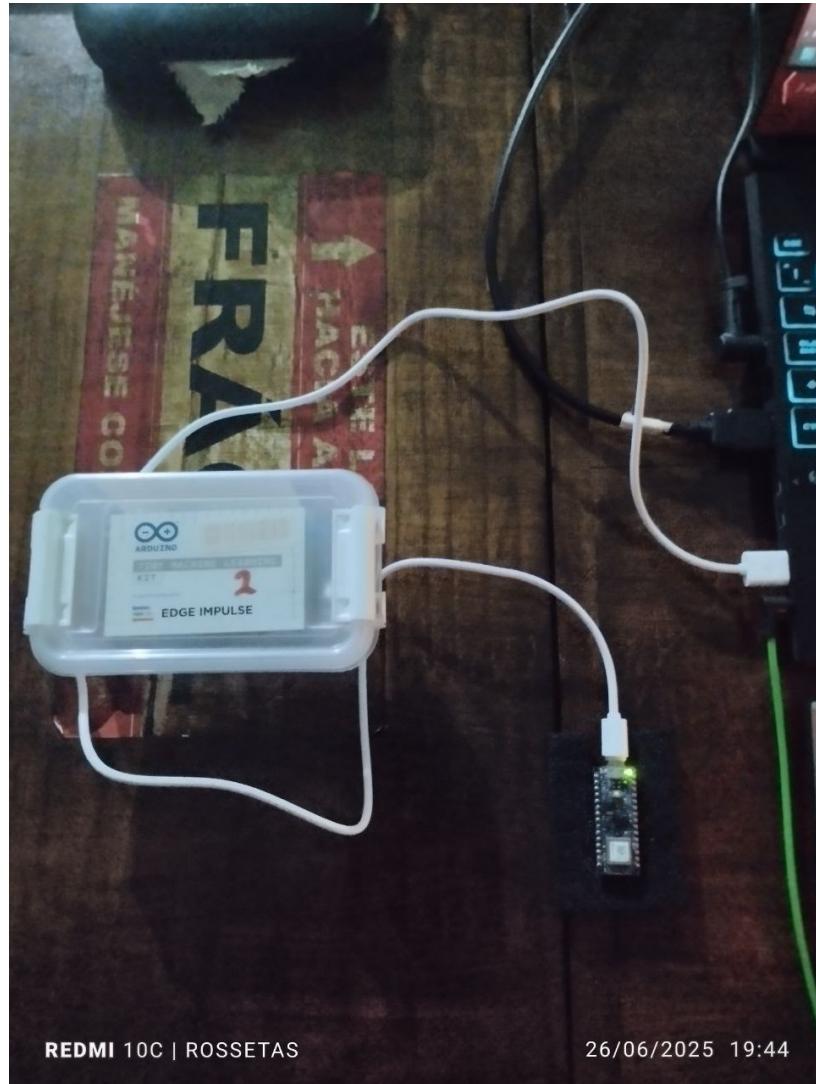


Figura 13: Circuito final

*Elaboración propia*

### 3.5. Análisis de resultados

Grabaremos los archivos de entrada en dos gráficos separados, aceleración y giroscopio, ya que cada conjunto de datos tiene diferentes unidades y escala.

#### 3.5.1. Movimiento: Círculos

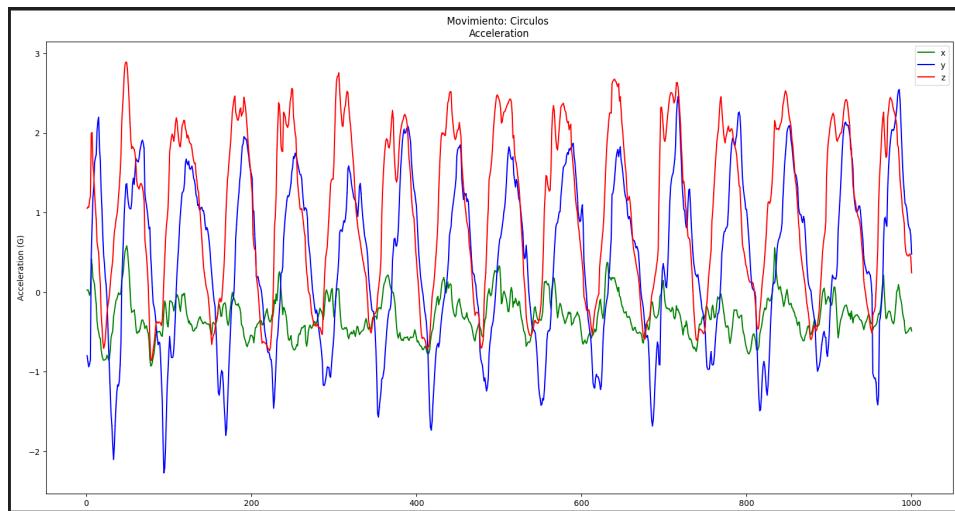


Figura 14: aceleración-círculos  
*Elaboración propia*

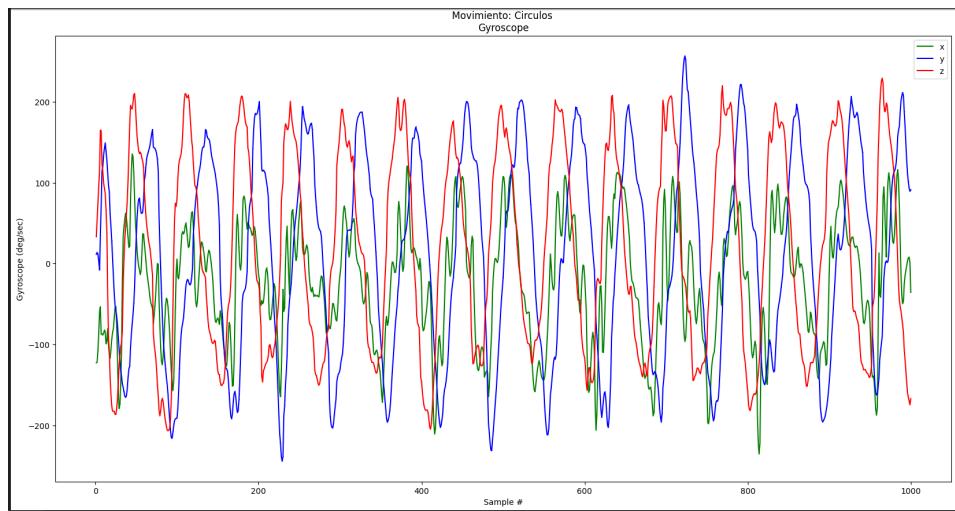


Figura 15: giroscopio-círculos  
*Elaboración propia*

### 3.5.2. Movimiento: Arriba-Abajo

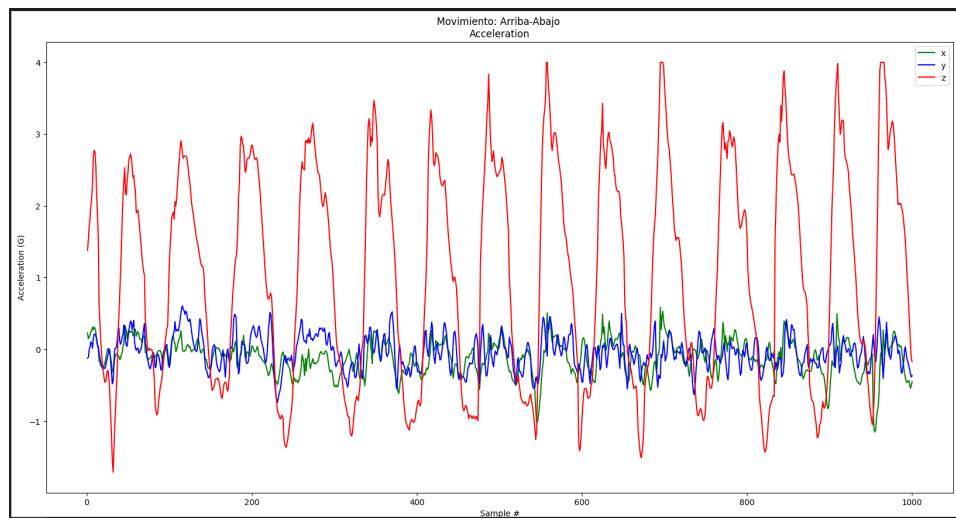


Figura 16: aceleración-Arriba\_Aabajo  
Elaboración propia

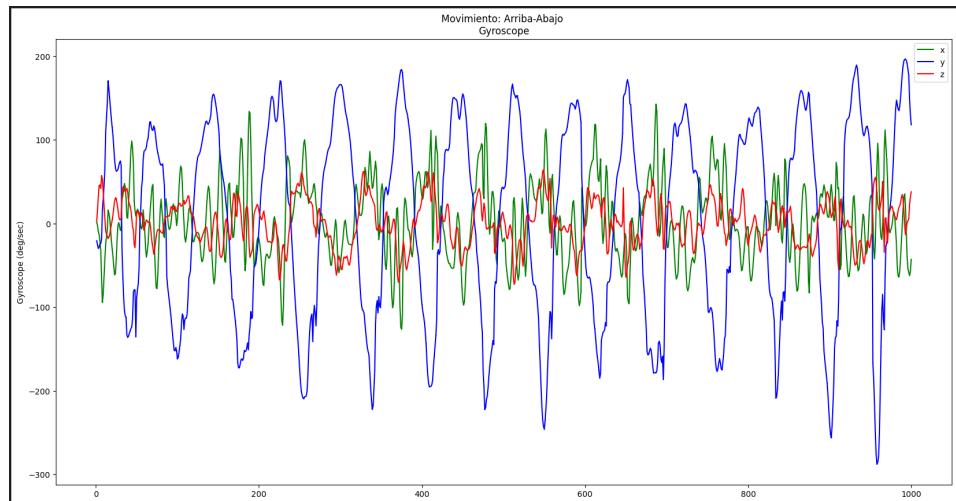


Figura 17: giroscopio-Arriba\_Aabajo  
Elaboración propia

### 3.5.3. Movimiento: Golpe

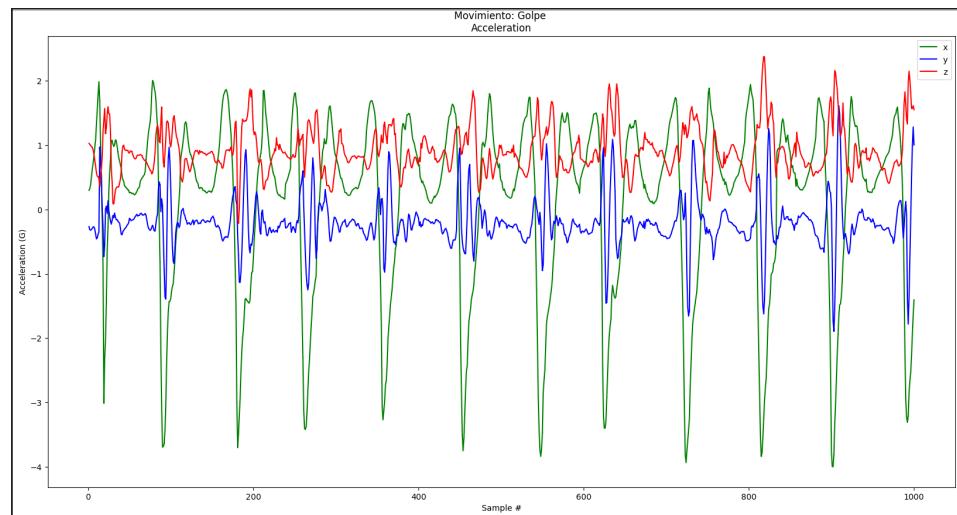


Figura 18: aceleración-golpe  
Elaboración propia

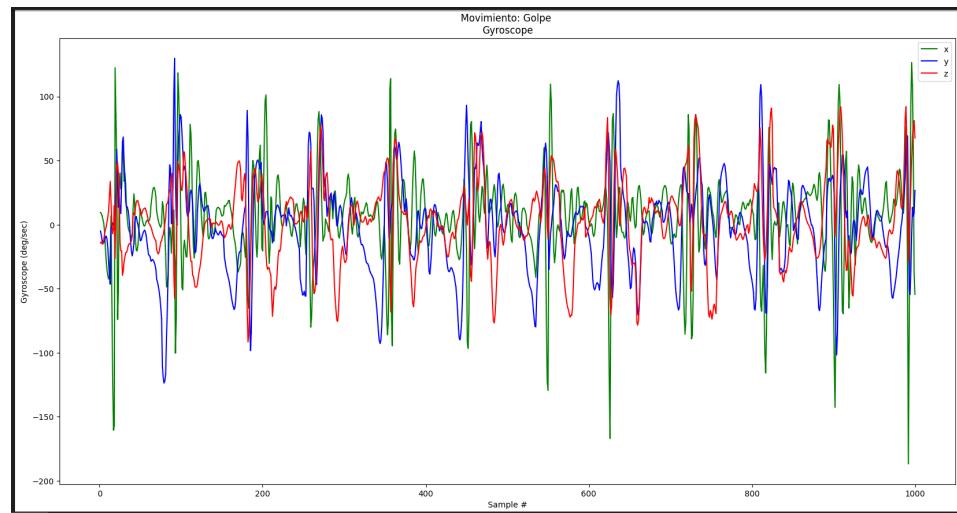


Figura 19: giroscopio-golpe  
Elaboración propia

### 3.5.4. Gráfica de la pérdida

Gráfica la pérdida para ver cuando el modelo deja de mejorar

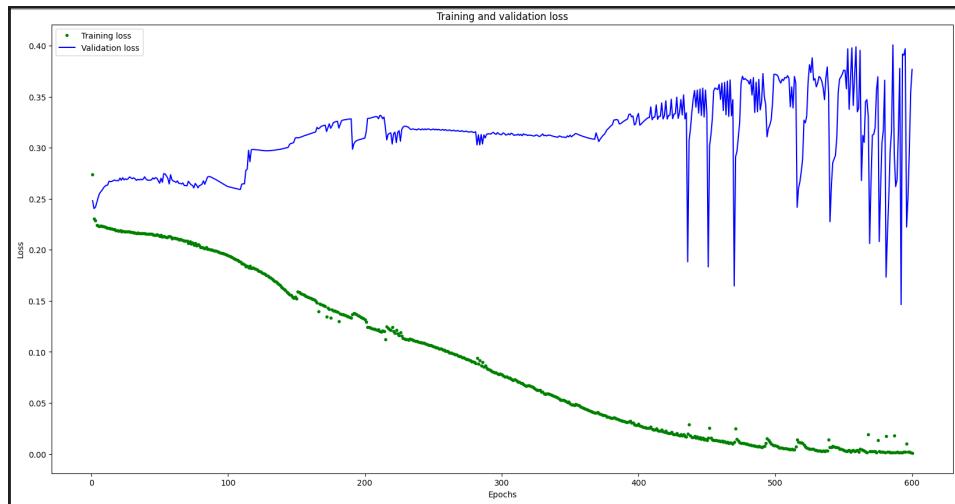


Figura 20: Graph the loss

*Elaboración propia*

### 3.5.5. Gráfico de la pérdida nuevamente, saltando un poco de inicio

Grabaremos los mismos datos que la celda de código anterior, pero comenzaremos en el índice 100 para que podamos acercarnos más una vez que el modelo comience a converger.

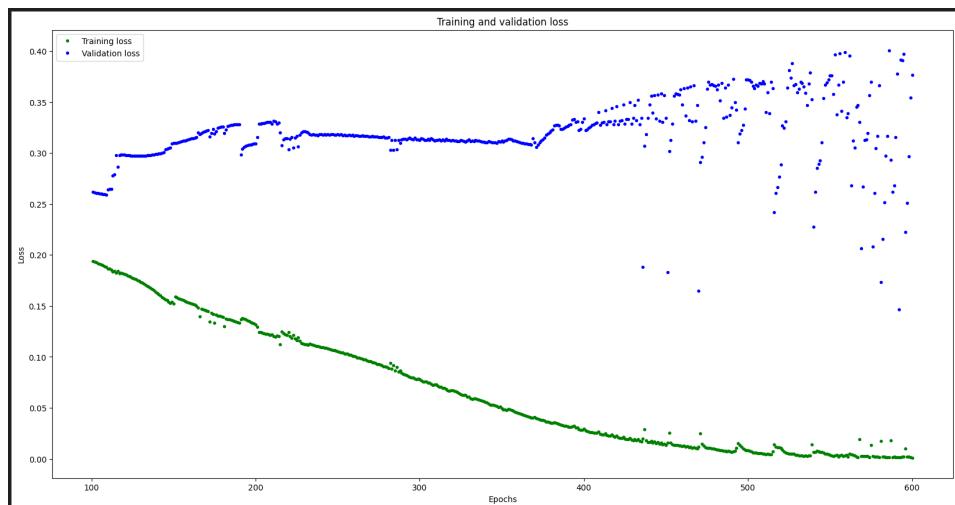


Figura 21: Graph the loss again, skipping a bit of the start

*Elaboración propia*

### 3.5.6. Gráfica el error absoluto medio

El ERROR ABSOLUTO MEDIO es otra métrica para juzgar el rendimiento del modelo.

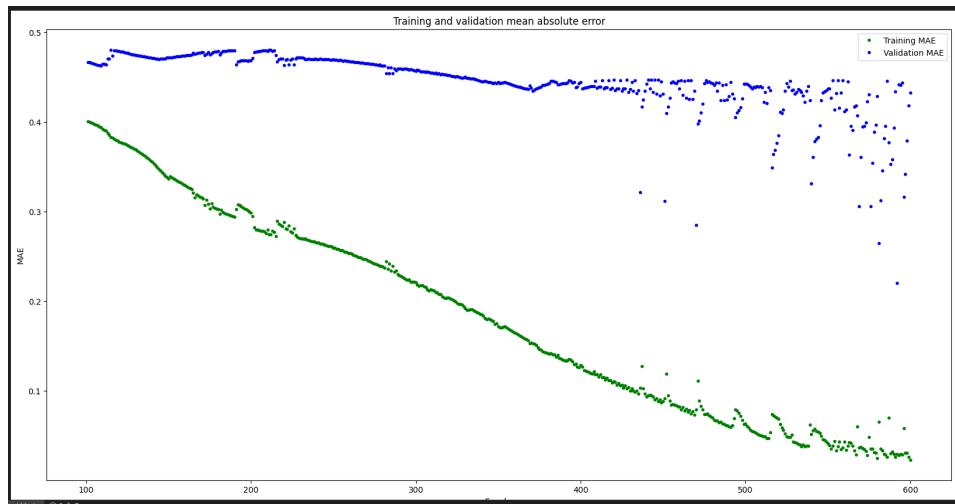


Figura 22: Graph the mean absolute error  
*Elaboración propia*

### 3.5.7. Trazar las predicciones junto con los datos de la prueba

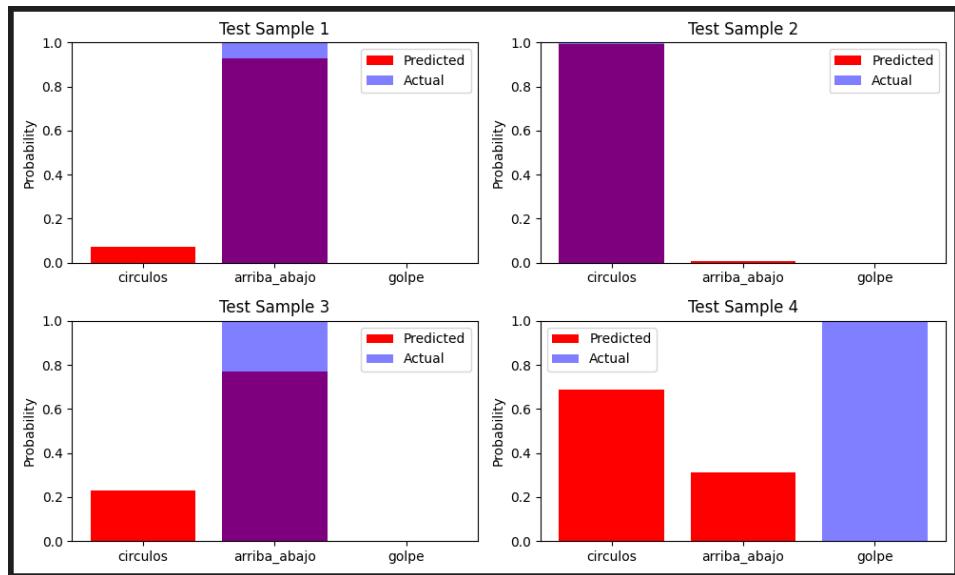


Figura 23: Plot the predictions along with to the test data  
*Elaboración propia*

### 3.6. Edge Impulse

#### 3.6.1. Data acquisition

En esta etapa se realizó la adquisición de datos provenientes del sensor IMU del Arduino Nano 33 BLE, capturando información de aceleración y giroscopio. Para cada tipo de movimiento definido —'círculos', 'arriba\_abajo' y 'golpe'— se recolectaron aproximadamente 1000 muestras mediante un script en Python. Estos datos fueron cargados en Edge Impulse, etiquetados adecuadamente y organizados para su posterior análisis. Esta fase es esencial para que el modelo aprenda a diferenciar correctamente entre cada gesto.

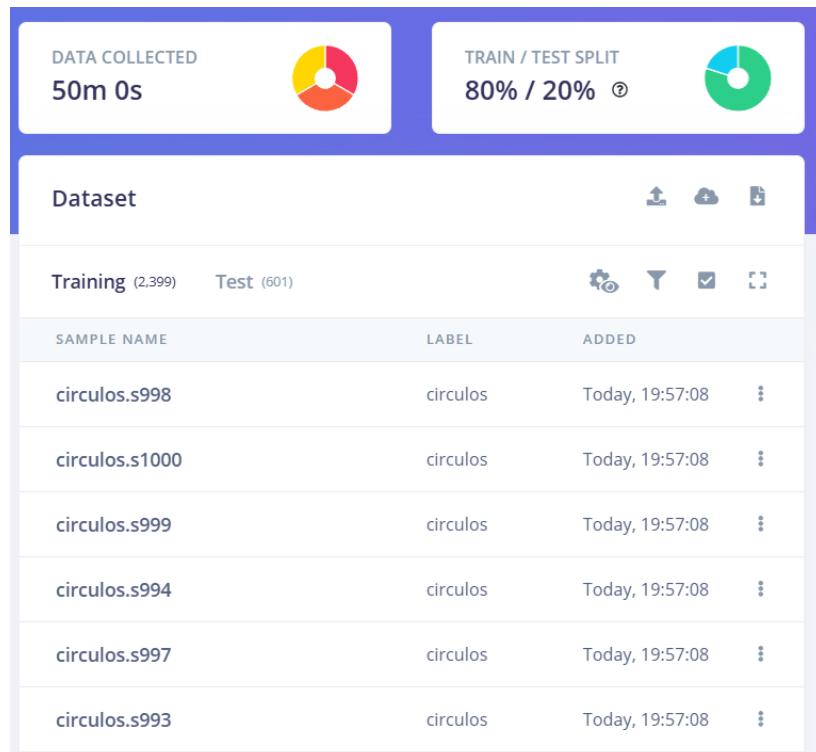


Figura 24: data  
*Elaboración propia*

#### 3.6.2. Create Impulse

En esta sección se diseñó la estructura del modelo de aprendizaje automático (impulso). Se configuró un bloque de 'Raw data' como entrada, seguido por un bloque de 'Classification', el cual entrena una red neuronal simple. El objetivo era crear un pipeline que tomara directamente los datos del IMU y los clasificara sin necesidad de extracción manual de características, manteniendo el modelo liviano para su ejecución en el microcontrolador.

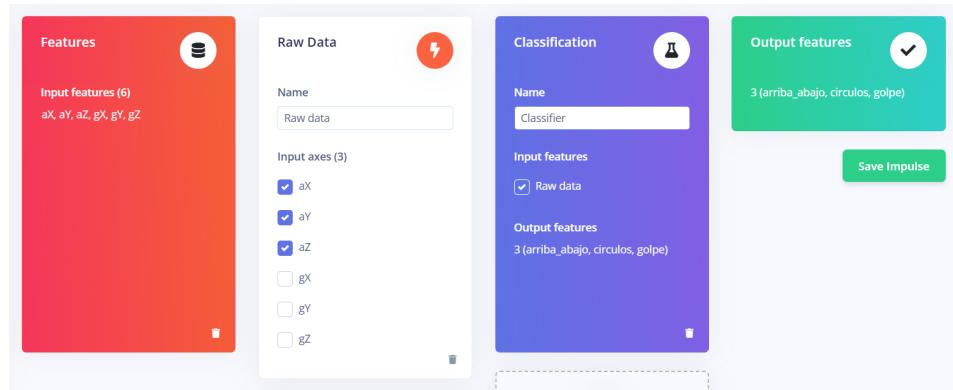


Figura 25: Impulse  
*Elaboración propia*

### 3.6.3. Raw data

Aquí se puede visualizar una representación gráfica de los datos brutos obtenidos de los sensores del IMU. Esta vista es útil para validar que las muestras estén correctamente etiquetadas y diferenciadas entre clases. Se observaron patrones distintivos para cada tipo de movimiento, lo que confirmaba que los datos eran adecuados para entrenamiento. Esta etapa es clave para detectar errores o ruidos antes de iniciar el modelado.

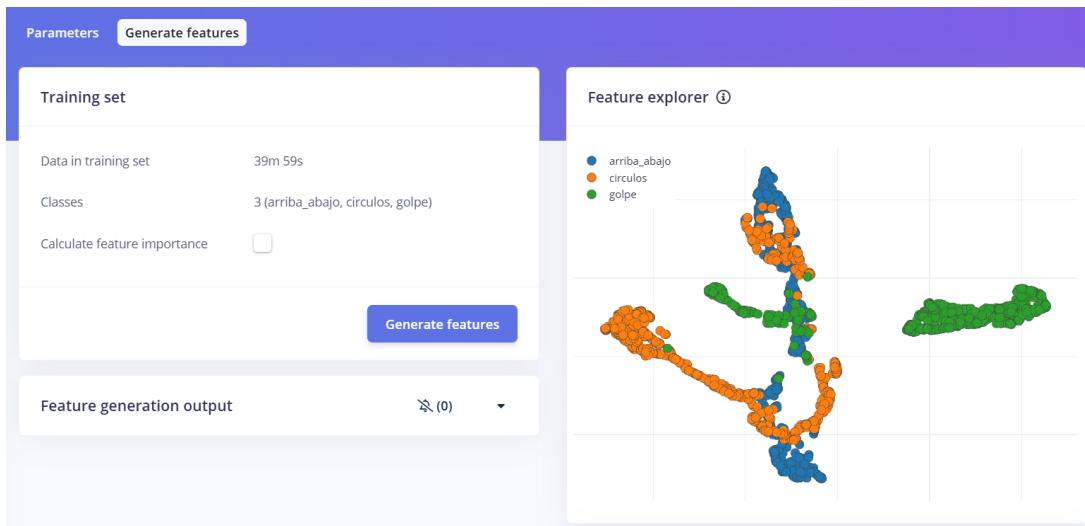


Figura 26: Features  
*Elaboración propia*

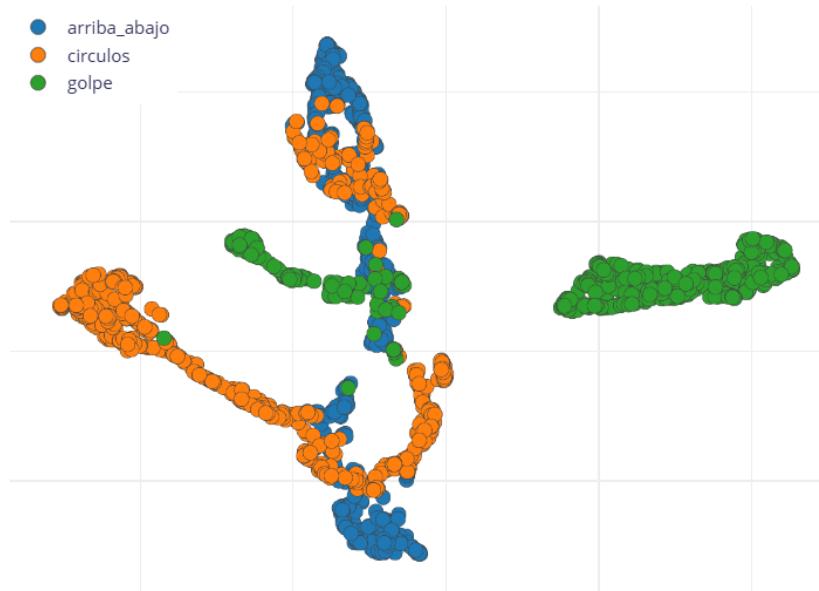


Figura 27: Features explorer  
*Elaboración propia*

### 3.6.4. Classifier

En esta fase se procedió a entrenar el modelo de red neuronal. Se utilizó una arquitectura simple con una capa densa oculta y una capa de salida con activación softmax. Como métrica de pérdida se utilizó categorical cross-entropy, y el optimizador elegido fue RMSprop, debido a su buen desempeño en problemas multiclase. Se dividieron los datos en: 60 % para entrenamiento, 20 % para validación y 20 % para prueba, logrando un modelo con buena precisión y baja sobreajuste.



Figura 28: Last training performance

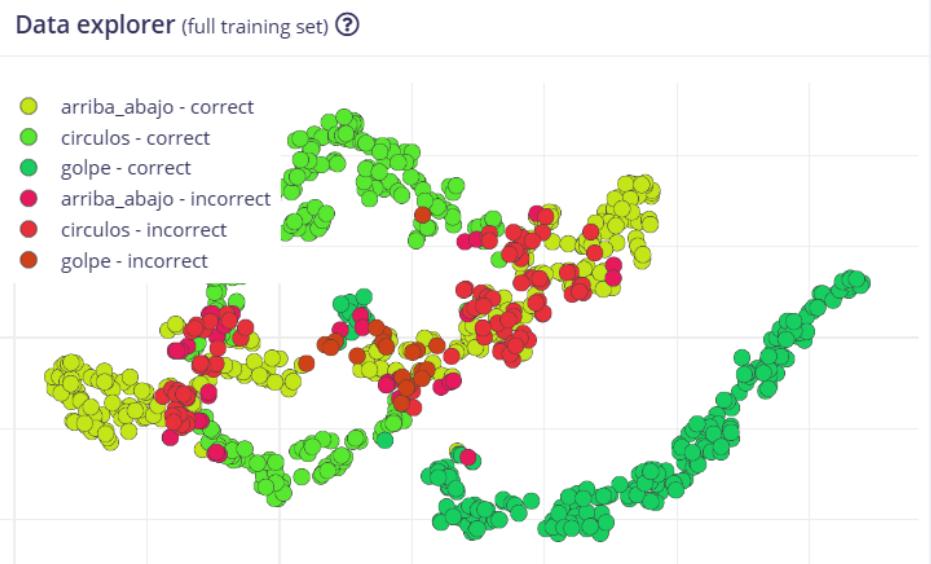
*Elaboración propia*

Figura 29: Data explorer

*Elaboración propia*

### 3.6.5. Model testing

Una vez entrenado el modelo, se procedió a la fase de pruebas del clasificador con datos no vistos. Se evaluaron los resultados utilizando la matriz de confusión generada por Edge Impulse. El modelo

fue capaz de reconocer correctamente los tres tipos de movimiento definidos, con niveles altos de precisión. Esta validación permitió verificar que el sistema general es confiable antes de exportarlo al dispositivo embebido.

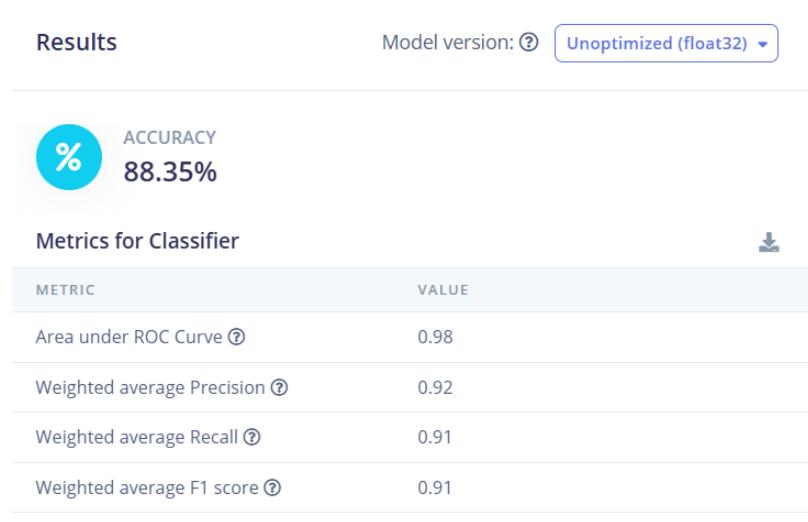


Figura 30: Results  
Elaboración propia

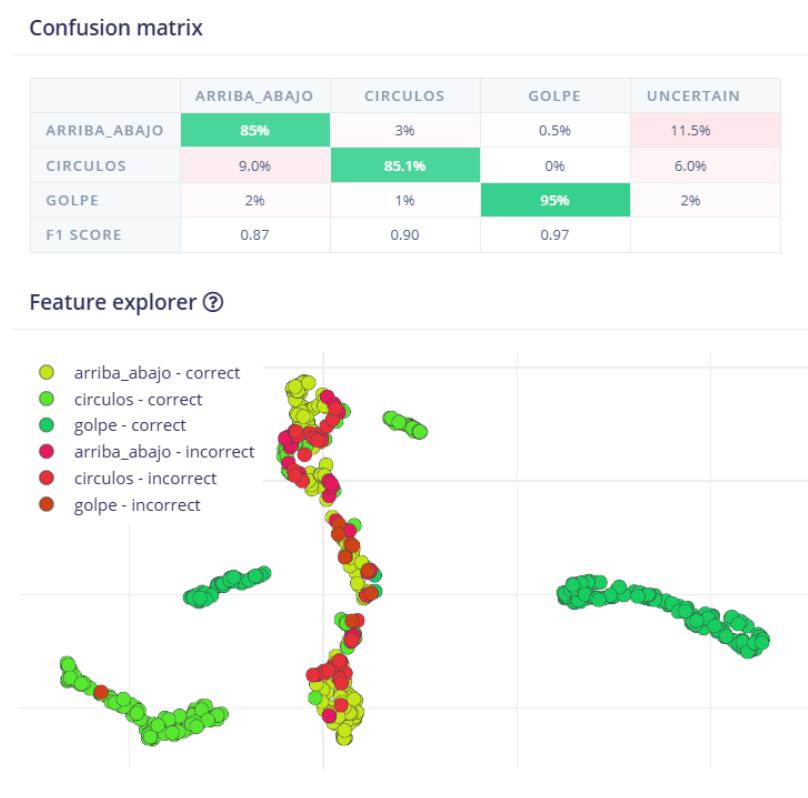


Figura 31: Results  
Elaboración propia

### 3.6.6. Deployment

Finalmente, se accedió a la sección de despliegue (Deployment), donde se exportó el modelo entrenado en formato de librería Arduino (.zip). Esta librería incluye el modelo en formato .tflite, junto con funciones necesarias para integrarlo en el entorno de desarrollo de Arduino. Se importó esta librería al IDE, y se desarrolló un programa que corre el modelo directamente en el Arduino Nano 33 BLE para detectar gestos en tiempo real. Los resultados de cada inferencia fueron enviados vía serial a una computadora para su registro.

#### Configure your deployment

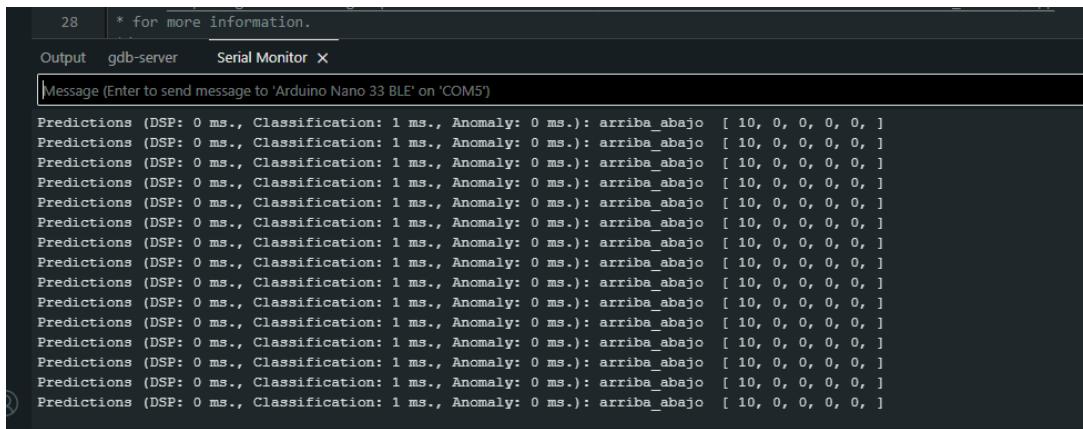
You can deploy your impulse to any device. This makes the model run without an internet connection, minimizes latency, and runs with minimal power consumption. [Read more.](#)

The screenshot shows the TensorFlow Model Optimizer interface. At the top, there's a search bar with 'Arduino library' and a clear button. Below it, a section titled 'SELECTED DEPLOYMENT' shows the 'Arduino library' selected, described as 'An Arduino library with examples that runs on most Arm-based Arduino development boards.' In the bottom section, 'MODEL OPTIMIZATIONS' is shown with the note that optimizations can increase performance but may reduce accuracy. A dropdown menu is open, showing 'TensorFlow Lite' selected. To the left, 'Quantized (int8)' is listed with a 'Selected' button. On the right, a table provides performance metrics:

	RAW DATA	CLASSIFIER	TOTAL
LATENCY	-	1 ms.	1 ms.
RAM	0.0K	3.0K	3.0K
FLASH	-	34.1K	-
ACCURACY			-

Figura 32: build  
Elaboración propia

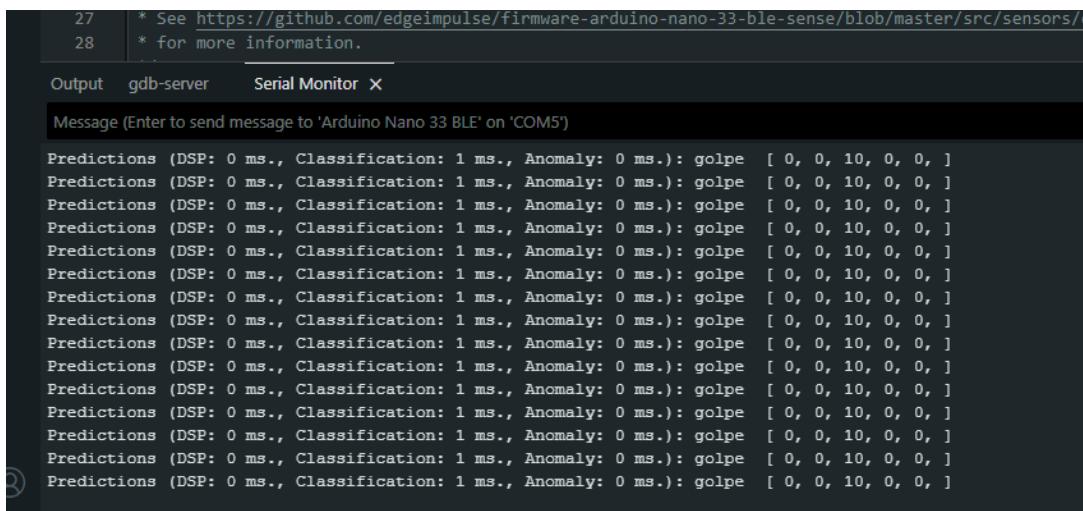
### 3.6.7. Resultados



The screenshot shows a Serial Monitor window with the following details:

- Line 28: \* for more information.
- Output tab is selected.
- Message input field: Message (Enter to send message to 'Arduino Nano 33 BLE' on 'COM5')
- Text content: Predictions (DSP: 0 ms., Classification: 1 ms., Anomaly: 0 ms.): arriba\_abajo [ 10, 0, 0, 0, 0, ]  
This line is repeated 15 times.

Figura 33: resultados-serial monitor

*Elaboración propia*


The screenshot shows a Serial Monitor window with the following details:

- Line 27: \* See <https://github.com/edgeimpulse/firmware-arduino-nano-33-ble-sense/blob/master/src/sensors/e>  
Line 28: \* for more information.
- Output tab is selected.
- Message input field: Message (Enter to send message to 'Arduino Nano 33 BLE' on 'COM5')
- Text content: Predictions (DSP: 0 ms., Classification: 1 ms., Anomaly: 0 ms.): golpe [ 0, 0, 10, 0, 0, ]  
This line is repeated 15 times.

Figura 34: resultados-serial monitor

*Elaboración propia*

## 4. Conclusiones y Recomendaciones

### 4.1. Conclusiones

- El **kit Arduino Nano 33 BLE Sense Lite** demostró ser una plataforma ágil para experimentar: la integración de IMU, BLE y otros sensores permite validar algoritmos directamente sobre hardware real.
- El uso de **Edge Impulse** agilizó todo el flujo de trabajo (captura, etiquetado, entrenamiento y despliegue), concentrando el esfuerzo en el diseño experimental más que en la infraestructura de software.
- La práctica ilustró los principios de **Tiny Machine Learning**: optimización de modelos, cuantización y ejecución en memoria restringida, acrediitando la capacidad de un microcontrolador con recursos ajustados para procesar y clasificar datos eficientemente.
- El estudio de **Reconocimiento de Actividad Humana (HAR)** sirvió como ejemplo práctico para comprender los fundamentos del aprendizaje automático aplicado a señales de movimiento, al relacionar gestos físicos con las salidas del modelo.

### 4.2. Recomendaciones

- Ajustar la sensibilidad de los sensores, identificando el punto de operación óptimo que garantice señales claras y consistentes.
- Explorar periféricos adicionales, como el micrófono o el sensor de color/proximidad, para enriquecer futuras aplicaciones con datos multisensoriales.
- En caso de querer una aplicación portátil, medir el consumo de energía para estimar la autonomía con batería.
- Expandir el proyecto implementando un contador de pasos, aprovechando la misma IMU para añadir una nueva funcionalidad sin aumentar la complejidad del hardware.

## Bibliografía

1. Material de clase
2. Arduino Nano 33 BLE Sense datasheets <https://docs.arduino.cc/hardware/nano-33-ble-sense/>
3. Edge Impulse documentation <https://docs.edgeimpulse.com/docs>

## 5. Apéndices

### 5.1. Repositorio Git

Repositorio GitHub: [https://github.com/Rossetas/MCU\\_Lab5](https://github.com/Rossetas/MCU_Lab5)