



DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACION

Opción 1.- TESIS

TRABAJO PROFESIONAL

**DESARROLLO DE UN SISTEMA NEUROMÓRFICO BASADO EN EL
RECONOCIMIENTO DE ACTIVIDADES EN ADULTOS MAYORES**

**QUE PARA OBTENER EL GRADO DE:
MAESTRO EN
INGENIERÍA ELECTRÓNICA**

PRESENTA:

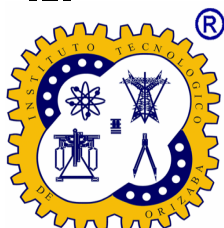
JUAN PABLO JARQUIN LARA

DIRECTOR DE TESIS:

DR. GERARDO ÁGUILA RODRÍGUEZ

CODIRECTOR DE TESIS:

NOMBRE DEL CODIRECTOR





Educación
Secretaría de Educación Pública



TECNOLÓGICO
NACIONAL DE MÉXICO



Instituto Tecnológico de Orizaba
División de Estudios de Posgrado

Orizaba, Veracruz, **19/noviembre/2025**
Dependencia: **División de Estudios de
Posgrado e Investigación**
Asunto: **Autorización de Impresión**
OPCION: I

C. JUAN PABLO JARQUIN LARA
Candidato(a) a Grado de Maestro(a) en:
INGENIERÍA ELECTRÓNICA
P R E S E N T E.-

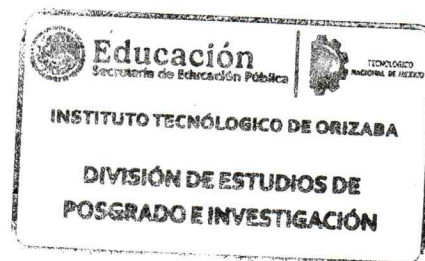
De acuerdo con el Reglamento de Titulación vigente de los Centros e Institutos Tecnológicos Federales del Tecnológico Nacional de México, de la Secretaría de Educación Pública, y habiendo cumplido con todas las indicaciones que la Comisión Revisora le hizo respecto a su Trabajo Profesional titulado:

**“DESARROLLO DE UN SISTEMA NEUROMÓRFICO BASADO EN EL RECONOCIMIENTO
DE ACTIVIDADES EN ADULTOS MAYORES”**

Comunico a Usted que este Departamento concede su autorización para que proceda a la impresión del mismo.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
CIENCIA – TÉCNICA - CULTURA®

DRA. OFELIA LANDETA ESCAMILLA
JEFA DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN



OG-13-F06



2025
Año de
**La Mujer
Indígena**

Av. Oriente 9 #852 Colonia Emiliano Zapata, C.P. 94320
Orizaba, Veracruz. Tel. 01 (272) 1105360
e-mail: info@orizaba.tecnm.mx www.orizaba.tecnm.mx





Instituto Tecnológico de Orizaba
División de Estudios de Posgrado

Orizaba, Veracruz, **19/noviembre/2025**
Asunto: **Revisión de trabajo escrito**

C. OFELIA LANDETA ESCAMILLA
JEFA DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN
PRESENTE.-

Los que suscriben, miembros del jurado, han realizado la revisión de la Tesis del (la) C.

JUAN PABLO JARQUIN LARA

La cual lleva el título de:

**“DESARROLLO DE UN SISTEMA NEUROMÓRFICO BASADO EN EL RECONOCIMIENTO DE
ACTIVIDADES EN ADULTOS MAYORES”**

Y concluyen que se acepta.

ATENTAMENTE
Excelencia en Educación Tecnológica®
CIENCIA – TÉCNICA - CULTURA®

PRESIDENTE: DR. GERARDO ÁGUILA RODRÍGUEZ

SECRETARIO: DR. OSCAR OSVALDO SANDOVAL
GONZÁLEZ

VOCAL: DRA. BLANCA ESTELA GONZÁLEZ
SÁNCHEZ

VOCAL SUP.: DR. IGNACIO HERRERA ÁGUILAR

FIRMA

FIRMA

FIRMA

FIRMA

TA-09-18



2025
Año de
**La Mujer
Indígena**

Av. Oriente 9 #852 Colonia Emiliano Zapata, C.P. 94320
Orizaba, Veracruz. Tel. 01 (272) 1105360
e-mail: info@orizaba.tecnm.mx www.orizaba.tecnm.mx



CONTENIDO

AGRADECIMIENTOS	13
RESUMEN	14
ABSTRACT	15
INTRODUCCIÓN	16
CAPÍTULO 1: FUNDAMENTACIÓN Y ANÁLISIS PRELIMINAR PARA EL DESA- ROLLO DE UN SISTEMA NEUROMÓRFICO ENFOCADO A HERRAMIENTAS PARA RECONOCIMIENTO DE ACTIVIDADES EN ADULTOS MAYORES	19
1.1 PLANTEAMIENTO DEL PROBLEMA.....	20
1.2 JUSTIFICACIÓN.....	21
1.3 HIPÓTESIS	21
1.4 OBJETIVOS	22
1.4.1 OBJETIVO GENERAL.....	22
1.4.2 OBJETIVOS ESPECÍFICOS	22
1.5 ESTADO DEL ARTE	22
1.5.1 INNOVACION EN EL DESARROLLO DE ALGORITMOS DE DEEP LEARNING PARA APLICACIONES BASADAS EN EL RECONOCIMIENTO DE ACTIVIDADES HUMANAS (HAR) MEDIANTE SENSORES INERCIALES	22
1.6 REVISIÓN TEÓRICA DE LOS SISTEMAS UTILIZADOS PARA EL DESA- ROLLO DE HERRAMIENTAS DE ASISTENCIA EN ADULTOS MAYORES 34	
1.6.1 ERGONOMÍA Y PORTABILIDAD DE LOS SISTEMAS WEARABLES ..	38
1.6.2 ESTUDIO COMPARATIVO DE LAS UNIDADES DE COMUNICACIÓN EN LOS SISTEMAS WEARABLES Y SUS CAPACIDADES	39

1.6.3	ANÁLISIS COMPARATIVO DE LOS SISTEMAS MICROCONTROLADORES PARA EL PROCESAMIENTO AUTOMÁTICO DE DATOS EN APLICACIONES WEARABLES	41
1.7	ASPECTOS GENERALES PARA EL DESARROLLO DE UN SISTEMA NEUROMÓRFICO.....	48
1.7.1	INTELIGENCIA ARTIFICIAL.....	48
1.7.2	MACHINE LEARNING Y DEEP LEARNING	49
1.7.3	REDES NEURONALES.....	51
1.7.3.1	REDES NEURONALES DE RETROALIMENTACIÓN	53
1.7.3.2	REDES NEURONALES CONVOLUCIONALES	53
1.7.3.3	REDES NEURONALES RECURRENTEES	53
1.7.4	REDES NEURONALES MULTICAPA.....	53
CAPÍTULO 2: DESCRIPCIÓN DE METODOLOGÍA PARA EL DESARROLLO DE UNA HERRAMIENTA DE ASISTENCIA BASADA EN UN SISTEMA NEUROMÓRFICO ORIENTADO AL RECONOCIMIENTO DE ACTIVIDADES EN ADULTOS MAYORES		55
CAPÍTULO 3: DESARROLLO EXPERIMENTAL PARA IMPLEMENTACIÓN DE UN SISTEMA NEUROMÓRFICO DE RECONOCIMIENTO DE ACTIVIDADES EN HERRAMIENTA DE ASISTENCIA PARA ADULTOS MAYORES		64
3.1	METODOLOGÍA PARA LA PROGRAMACIÓN Y MONTAJE DE UN MICROCONTROLADOR ATMEGA328P-AU (Etapa 1)	64
3.1.1	Revisión y selección de microcontrolador.....	65
3.1.2	Prueba de sistema mínimo en Protoboard	66
3.1.3	Manejo de tecnología SMD	69

3.1.4	Diseño, manufactura y montaje del sistema embebido para microcontrolador ATmega328P-AU	71
3.2	METODOLOGÍA PARA EL DESARROLLO DE INTERFAZ CAPACITIVA (Etapa 2)	72
3.2.1	Comprensión del fenómeno	73
3.2.2	Análisis de diseño y selección de componentes	77
3.2.3	Diseño y desarrollo de interfaces	79
3.2.4	Manufactura, montaje y pruebas del sistema embebido de procesamiento de señales capacitivas	80
3.3	METODOLOGÍA PARA EL DESARROLLO DE UN SISTEMA EMBEBIDO DE COMUNICACIÓN GSM (Etapa 3).....	84
3.3.1	Revisión y análisis de las características técnicas del módulo de comunicación GSM	85
3.3.2	Selección de componentes y pruebas preliminares	86
3.3.3	Diseño y ajustes de código mediante comandos AT.....	90
3.3.4	Manufactura, montaje y pruebas del sistema embebido para comunicación GSM.....	93
3.4	METODOLOGÍA PARA EL DESARROLLO DE SISTEMA EMBEBIDO PARA REGISTRO DE DATOS INERCIALES (Etapa 4).....	94
3.4.1	Análisis preliminar de especificaciones de la Unidad de Medición Inercial (IMU)	95
3.4.2	Configuración de la Unidad de Medición Inercial	96
3.4.3	Pruebas preliminares y ajuste del código para monitoreo	97

3.4.4	Diseño, manufactura y validación del sistema embebido de monitoreo inercial	97
3.5	METODOLOGÍA PARA EL DESARROLLO DE UNA RED NEURONAL PARA INFERENCIA INERCIAL (Etapa 5).....	108
3.5.1	Análisis teórico para ubicación del sistema embebido de monitoreo inercial.....	109
3.5.2	Selección de movimientos y captura de base de datos para entrenamiento....	111
3.5.3	Postprocesamiento de los datos registrados	113
3.5.4	Derarrollo del algoritmo para la creación y entrenamiento del modelo de Red Neuronal	121
3.6	METODOLOGÍA PARA LA INTEGRACIÓN E IMPLEMENTACIÓN DE UNIDADES DEL SISTEMA NEUROMÓRFICO (Etapa 6).....	124
3.6.1	Implementación del modelo de Red Neuronal en microcontrolador	125
3.6.2	Diseño y manufactura del sistema embebido final	129
3.6.3	Integración del sistema neuromórfico en Wearable	133
3.6.4	Pruebas y validación	135
CAPÍTULO 4: PRESENTACIÓN Y DISCUSIÓN DE LOS RESULTADOS		137
4.1	Gráficos de registro de actividades inerciales	138
4.2	Modelo de Red Neuronal Multicapa por sumatoria de valores absolutos.....	141
4.3	Correlación de variables	148
4.4	Modelo de selección de características por XGBoost.....	161
4.5	Integración del sistema en herramienta de asistencia wearable.....	170
4.6	Tiempo de respuesta del sistema.....	171
4.7	Consumo y eficiencia energética del sistema.....	174

CONCLUSIONES	178
REFERENCIAS BIBLIOGRÁFICAS	180
ANEXOS	184
4.8 ANEXO A	184
4.9 ANEXO B	187
4.10 ANEXO C	189
4.11 ANEXO D	195

LISTA DE FIGURAS

Figura 1.1.	Esquema representativo de la evolución de la Inteligencia Artificial.	48
Figura 1.2.	Esquema representativo de la evolución de la Inteligencia Artificial.	52
Figura 2.1.	Metodología para el desarrollo del proyecto de investigación etapa 1.	55
Figura 2.2.	Metodología para el desarrollo del proyecto de investigación etapa 1.	56
Figura 2.3.	Metodología para el desarrollo del proyecto de investigación etapa 2.	57
Figura 2.4.	Metodología para el desarrollo del proyecto de investigación etapa 3.	58
Figura 2.5.	Metodología para el desarrollo del proyecto de investigación etapa 4.	59
Figura 2.6.	Metodología para el desarrollo del proyecto de investigación etapa 5.	60
Figura 2.7.	Metodología para el desarrollo del proyecto de investigación etapa 6.	62
Figura 3.1.	Diagrama esquemático de conexiones para programación del microcontrolador ATmega328P-AU	67
Figura 3.2.	Conexión de sistema mínimo para carga de Bootloader de microcontrolador ATmega328P-AU.	68
Figura 3.3.	Conexión de sistema mínimo para carga de código en microcontrolador ATmega328P-AU.	69
Figura 3.4.	a) Tarjeta de circuito impreso en sustrato de cobre de Cupón de componentes de montaje superficial (SMD) de empaquetado 0805. b) Cupón componentes SMD 0805 con solder mask ultravioleta.	70
Figura 3.5.	Diseño de tarjeta de circuito impreso para microcontrolador ATmega328P-AU en Ultiboard 14.0.	71
Figura 3.6.	Diagrama esquemático de sensor capacitivo por descompensación de carga mediante oscilador NE555.	74
Figura 3.7.	Diseño para manufactura de circuito de sensor capacitivo por descompensación de carga en Ultiboard 14.0.	75
Figura 3.8.	Pruebas de manufactura de tarjetas de circuito impreso de sensor capacitivo por descompensación de carga.	76
Figura 3.9.	Prototipo de sensor táctil basado en descompensación de carga.	76
Figura 3.10.	Estándar de geometrías para el diseño pads capacitivos. [31]	78
Figura 3.11.	a) Interfaz capacitiva matricial para sensor de capacitancia mutua. b)	

Interfaz capacitiva con pads con forma cuadrada para autocapacitancia. c) Interfaz capacitiva con pads con forma circular para autocapacitancia. d) Interfaz de autocapacitancia para interconexión con interfaz visual de QTDesigner.	79
Figura 3.12. Sistema mínimo para el sensor capacitivo AT42QT1070.....	80
Figura 3.13. Sistema mínimo para el sensor capacitivo AT42QT1070 versión 2.....	81
Figura 3.14. Diagrama esquemático de sistema embebido para procesamiento de señales capacitivas de AT42QT1070.	82
Figura 3.15. Diseño de tarjeta de circuito impreso de sistema embebido para procesamiento de señales capacitivas de AT42QT1070.	83
Figura 3.16. Sistema embebido para pruebas de sensor capacitivo AT42QT1070.....	83
Figura 3.17. Distribución de pines de conexión del módulo SIM800L CITAR IMA-GEN.....	86
Figura 3.18. Prueba de módulo SIM800L para enviar mensajes de texto SMS mediante microcontrolador ATmega328P	87
Figura 3.19. Prueba experimental de módulo SIM800L para llamadas telefónicas mediante microcontrolador ATmega328P.....	88
Figura 3.20. Prueba experimental de módulo SIM800L para recepción de mensajes de texto mediante microcontrolador ATmega328P.a) Envío de la palabra ON vía SMS al módulo para encender LED. b) Envío de la palabra OFF vía SMS al módulo para apagar LED.	89
Figura 3.21. Prueba experimental de sensor de movimiento inercial integrado con módulo de comunicación GSM.	90
Figura 3.22. Prueba experimental de periféricos de audio integrados con módulo de comunicación GSM.	91
Figura 3.23. Prueba experimental de periféricos de audio integrados con módulo de comunicación GSM.	93
Figura 3.24. Tarjeta de circuito impreso de sistema embebido para comunicación GSM.....	94
Figura 3.25. Código para analizar el comportamiento del giroscopio del sensor MPU6050 en °/s configurado a ± 250 °/s.	97
Figura 3.26. Diagrama esquemático para sistema embebido para pruebas de sensor de	

movimiento inercial.....	98
Figura 3.27. Diseño de sistema embebido para pruebas de sensor de movimiento inercial.....	98
Figura 3.28. Revelado de tarjeta de circuito impreso del sistema embebido para pruebas de sensor de movimiento inercial.....	99
Figura 3.29. Sistema embebido para pruebas de sensor de movimiento inercial.....	99
Figura 3.30. Esquema general de prueba experimental para configuración del sistema embebido de monitoreo inercial.....	100
Figura 3.31. Base de soporte para sistema embebido de monitoreo inercial.....	101
Figura 3.32. Esquema general de prueba experimental para configuración del sistema embebido de monitoreo inercial.....	102
Figura 3.33. Esquema general de los componentes fundamentales para la transmisión de datos inerciales a la herramienta de registro.....	103
Figura 3.34. Interfaz de registro de datos inerciales de herramienta PLX-DAQ en Excel 2016.....	104
Figura 3.35. Gráfica 1 de Clustering por K-Means para 20 grupos de datos de aceleración resultante en caída libre.....	105
Figura 3.36. Gráfica 2 de Clustering por K-Means para 20 grupos de datos de aceleración resultante en caída libre.....	107
Figura 3.37. Representación y orientación de ejes de sensor MPU6050.....	109
Figura 3.38. Representación de la colocación del sistema embebido de HAR como herramienta de asistencia.....	110
Figura 3.39. Estructura de los bloques de datos registrados en la herramienta PLX-DAQ de Excel.....	114
Figura 3.40. Diagrama esquemático del sistema embebido final para reconocimiento de actividades humanas en adultos mayores.....	131
Figura 3.41. Diseño de tarjeta de circuito impreso del sistema embebido final para reconocimiento de actividades humanas en adultos mayores.....	132
Figura 3.42. a) Diseño de base del enclosure para el sistema embebido. b) Diseño de tapa del enclosure para el sistema embebido. c) Vista isométrica del enclosure del sistema embebido.....	133

Figura 3.43. Diseño de base del enclosure para el sistema embebido	134
Figura 3.44. Integración del sistema neuromórfico en enclosure para herramienta de asistencia wearable.....	135
Figura 4.1. Registro de un evento crítico Caída "mediante la UMI. a) Datos iniciales de aceleración b) Datos iniciales de velocidad angular.....	139
Figura 4.2. Registro de un evento crítico Caída "mediante la UMI. a) Datos reducidos de aceleración b) Datos reducidos de velocidad angular.....	140
Figura 4.3. Gráfico representativo de pérdida en la fase de entrenamiento y fase de validación del modelo.	143
Figura 4.4. Matriz de confusión para validación del modelo de Red Neuronal.....	145
Figura 4.5. Gráfico de dispersión de las mediciones reales en la fase de prueba.	146
Figura 4.6. Gráfico de dispersión del Clustering por K-means en la fase de prueba.	147
Figura 4.7. Gráfico de Indices de Correlación de Pearson para la clase Agacharse.	149
Figura 4.8. Gráfico de Indices de Correlación de Pearson para la clase Caída.....	150
Figura 4.9. Gráfico de Indices de Correlación de Pearson para la clase Caminar Lento.	151
Figura 4.10. Gráfico de Indices de Correlación de Pearson para la clase Caminar Normal.....	152
Figura 4.11. Gráfico de Indices de Correlación de Pearson para la clase Caminar Rápido.	153
Figura 4.12. Gráfico de Indices de Correlación de Pearson para la clase Levantarse Lento.	154
Figura 4.13. Gráfico de Indices de Correlación de Pearson para la clase Levantarse Normal.....	155
Figura 4.14. Gráfico de Indices de Correlación de Pearson para la clase Levantarse Rápido.	156
Figura 4.15. Gráfico de Indices de Correlación de Pearson para la clase Sentarse Lento.....	158
Figura 4.16. Gráfico de Indices de Correlación de Pearson para la clase Sentarse Normal.....	159
Figura 4.17. Gráfico de Indices de Correlación de Pearson para la clase Sentarse Rápido.	160
Figura 4.18. Gráfico representativo de pérdida en la fase de entrenamiento y fase de validación del modelo por XGBoost.....	166

Figura 4.19. Matriz de confusión para validación del modelo de Red Neuronal de modelo XGBoost.	168
Figura 4.20. Integración del sistema neuromórfico en la prenda vestible (wearable) con sus partes.	171

LISTA DE TABLAS

Tabla 1.1.	Comparativa de módulos de comunicación utilizados en sistemas embebidos.	40
Tabla 1.2.	Comparativa de microcontroladores utilizados en sistemas embebidos.	43
Tabla 1.3.	Comparativa de sensores capacitivos de la familia AT42QT.	45
Tabla 1.4.	Comparativa de sensores inerciales utilizados en sistemas embebidos.	47
Tabla 3.1.	Configuraciones de filtros RC para las salidas de audio del módulo GSM	92
Tabla 3.2.	Sensibilidad del acelerómetro del MPU6050.	96
Tabla 3.3.	Sensibilidad del giroscopio del MPU6050.	96
Tabla 3.4.	Medición del peso de la base de soporte para el sistema embebido de monitoreo inercial.	103
Tabla 3.5.	Índice de clases del modelo para el sistema neuromórfico	128
Tabla 4.1.	Estructura final del modelo de red neuronal.	141
Tabla 4.2.	Reporte de clasificación del modelo en la fase de validación.	142
Tabla 4.3.	Importancia de las 50 características más relevantes del modelo XGBoost	162
Tabla 4.4.	Importancia de las 50 características más relevantes del modelo XGBoost	163
Tabla 4.5.	Estructura final del modelo de red neuronal por XGBoost para 50 características más importantes.	164
Tabla 4.6.	Reporte de clasificación del modelo en la fase de validación.	165
Tabla 4.7.	Resultados de tiempo de inferencia y envío del sistema neuromórfico	172
Tabla 4.8.	Tiempos de respuesta del sistema neuromórfico para envío de alertas.	173
Tabla 4.9.	Consumo estimado de corriente de cada componente del sistema neuromórfico.	175
Tabla 4.10.	Estimación de autonomía de la batería de 2500 mAh (3.7 V) en distintos escenarios.	176
Tabla 4.11.	Niveles de señal (CSQ) y su equivalencia en dBm del módulo SIM800L.	186

LISTA DE ECUACIONES

Ecuación 3.1.	Estructura de los datos inerciales	115
Ecuación 3.2.	Fórmula para el número de "n" mediciones.....	115
Ecuación 3.3.	Fórmula para el cálculo del peso o energía del bloque de datos	115
Ecuación 3.4.	Orden para la selección de los bloques de datos más representativos.	116
Ecuación 3.5.	Estructura de la nueva muestra de 50 bloques de datos más representati- vos.	116
Ecuación 3.6.	Cálculo de datos totales por cada muestra o medición de la clase.	116
Ecuación 3.7.	Conjunto de datos de entrenamiento	117
Ecuación 3.8.	Ecuación para predicción por suma de árboles de decisión.....	117
Ecuación 3.9.	Ecuación de función objetivo para cada iteración t	118
Ecuación 3.10.	Término de regularización del árbol.....	118
Ecuación 3.11.	Aproximación de segundo orden (Taylor)	118
Ecuación 3.12.	Fórmulas del gradiente y la segunda derivada (hessiana) de la función de pérdida.....	119
Ecuación 3.13.	Ecuación para el cálculo de ganancia en nodo de árbol	119
Ecuación 3.14.	Ecuaciones para el cálculo de gradientes y hessianas de cada lado de partición.....	119
Ecuación 3.15.	Expresión para el cálculo de la importancia total de una característica j	120
Ecuación 3.16.	Expresión para el cálculo de la importancia promedio de cada caracte- rística considerando distintos modelos.....	120
Ecuación 3.17.	Fórmula para el ordenamiento de las características según su impor- tancia.....	120
Ecuación 3.18.	Fórmula para la selección de las características N más relevantes.....	120
Ecuación 3.19.	Representación de la construcción de la matriz de datos seleccionadas por XGBoost.....	121
Ecuación 3.20.	Fórmula para la inferencia de clase en microcontrolador.	128

AGRADECIMIENTOS

A mis compañeros:

A ustedes que fueron mis acompañantes durante esta aventura que vivimos con alegría, algunas veces con preocupaciones, y otras veces atravesando por adversidades pero todo el tiempo siempre uno apoyando al otro y no dejando que cayera. Quiero agradecerles por mantener siempre el vínculo de apoyo entre esta familia que formamos para que esta etapa tan importante de nuestra vida fuera tan extraordinaria.

A mi familia:

Mi familia ha sido siempre el pilar de todas mis metas y logros, siempre estaré agradecido por todos esos consejos y por ese apoyo que incondicionalmente me brindaron y confiaron en mí.

A mis padres:

Ustedes han sido siempre mi motor de vida que me impulsa cada día a seguir adelante, a luchar por mis sueños y metas, así como a nunca rendirme ante cualquier adversidad. Muchas gracias por todo el apoyo incondicional que me brindaron a lo largo de esta etapa y agradezco que se encuentren presentes en este momento tan importante para mí.

A mi hermana:

A ti que, desde que fuimos pequeños me viste crecer, me viste equivocarme y en algunas ocasiones me corregiste, te quiero agradecer por ese apoyo y ese amor que siempre me tuviste, todos esos consejos que me diste y las pláticas que me diste cuando más las necesitaba, después de todo lo que hemos vivido quiero que sepas que siempre has sido mi ejemplo a seguir y que admiro la persona que eres y la persona en la que te convertirás.

RESUMEN

En este trabajo se presenta el diseño, manufactura e implementación de un sistema neuromórfico embebido orientado al reconocimiento de actividades humanas y la detección de caídas en adultos mayores. El sistema emplea una unidad de medición inercial (IMU) para registrar señales de aceleración y velocidad angular, las cuales son procesadas mediante un modelo de Deep Learning basado en redes neuronales multicapa (MLP).

Se desarrolló una metodología de seis etapas que abarca la adquisición de datos, el preprocesamiento de señales, la selección de características, el entrenamiento del modelo y su implementación en un microcontrolador de recursos limitados. Los resultados experimentales muestran que el modelo MLP por sumatoria de valores absolutos presenta el mejor desempeño, logrando inferencias en tiempo real con tiempos del orden de milisegundos y un consumo energético adecuado para operación autónoma.

El sistema final demuestra ser una herramienta viable para el monitoreo continuo y la detección oportuna de caídas, con potencial para futuras mejoras en conectividad, miniaturización y autonomía.

ABSTRACT

This work presents the design, development, and implementation of an embedded neuromorphic system for human activity recognition and fall detection in older adults. The system employs an Inertial Measurement Unit (IMU) to acquire acceleration and angular velocity signals, which are processed using a Multilayer Perceptron (MLP)–based Deep Learning model to identify the characteristic kinematic patterns of each activity.

A six-stage methodology was developed, encompassing data acquisition, signal preprocessing, feature selection, model training, and real-time implementation on a resource-constrained microcontroller. Experimental results show that the MLP model using the summation of absolute values achieves the best balance between accuracy, computational efficiency, and ease of deployment. The embedded system delivers real-time inference with millisecond-level response times and an energy consumption profile suitable for autonomous operation.

The final prototype demonstrates the feasibility of the proposed neuromorphic system as a practical tool for continuous monitoring and timely fall detection, offering a foundation for future improvements in connectivity, hardware miniaturization, and incremental model training.

INTRODUCCIÓN

Un tema de gran relevancia a nivel nacional vinculado con la salud, es el envejecimiento de la población. En México, la esperanza de vida es de 76 años en promedio [1]. Basta con recordar que en 1950 había 5 personas mayores (60 años o más) por cada 100 habitantes y para 2021 la cifra había llegado a 12. De mantenerse esta tendencia, la proyección indica que en 2050 serán 23 por cada 100 personas adultas mayores [2]. Esto representa un aumento de 13.7 millones de adultos mayores de 71 años, tomando como referencia inicial la población de 1950, y según las estimaciones se espera que para 2050 la población de este grupo alcanzará los 33.4 millones. Así mismo, se reportó que en 2011 el número de adultos mayores con alguna dependencia física fue de 1.2 millones, la cual aumentará significativamente entre 2021 y 2025.

Entre las afecciones comunes de la vejez, la ciencia médica reporta la pérdida de audición y la visión, problemas articulares, problemas emocionales y disminución de la calidad de vida generada por el síndrome de caídas en adultos mayores. A medida que los seres humanos envejecen, aumenta la probabilidad de experimentar varias afecciones al mismo tiempo y con mayor impacto.[3]

Actualmente existen herramientas en el mercado que brindan soluciones adecuadas a las diferentes necesidades del adulto mayor, tales como sillas de ruedas, andadores y bastones que ayudan a mejorar la movilidad del usuario. Por otra parte, existen también herramientas sofisticadas que monitorean variables fisiológicas, tales como el nivel de glucosa, la oxigenación en la sangre, la presión arterial, entre otras.

Sin embargo, otras necesidades, tales como la comunicación automática, la asistencia ante una caída inesperada o el simple monitoreo de la actividad humana (ponerse de pie, caminar, sentarse, agacharse) , son esenciales en la vejez, ya que hasta el momento no se encuentra en el mercado una herramienta confiable que las satisfaga.

En consecuencia, en este trabajo de investigación se propone desarrollar un sistema neuromórfico como herramienta de asistencia para adultos mayores, con funciones de reconocimiento de actividades humanas, comunicación oral y asistencia ante eventos inesperados.

En vista de lo anterior, el objetivo principal que hemos planteado para desarrollar este trabajo consiste en diseñar, manufacturar e implementar arreglos de sensores capacitivos con diferentes geometrías que permitan implementar funciones tales como la detección de caídas y la compensación de la movilidad, enfocándonos en las necesidades de los adultos mayores. En particular, se ha puesto especial atención en utilizar los procedimientos normalmente aceptados para el desarrollo de sistemas embebidos, con el fin de obtener una unidad de sensado de variables (posición, aceleración o contacto físico), la unidad de comunicación GSM (Global System for Mobile Communications), la unidad de suministro de potencia eléctrica y, como elemento clave de la aplicación, la unidad de reconocimiento automático de la actividad humana (sistema neuromórfico). Como complemento de este trabajo de investigación, se analizan los sensores capacitivos como una posible integración para el reconocimiento gestual para establecer la comunicación GSM; sin embargo, cabe resaltar que es un módulo experimental.

Con el objetivo de hacer comprensible este trabajo, lo hemos estructurado en cuatro capítulos. En el capítulo uno se presenta la fundamentación y el análisis preliminar del presente trabajo de investigación, dando a conocer la problemática, la justificación, los objetivos, la hipótesis y una sección de marco teórico en la que se basa el desarrollo del trabajo. De la misma manera, se presenta una sección de Estado del Arte para contextualizar los avances científicos en el campo que abarca nuestra propuesta de solución.

En el capítulo dos se presenta la metodología experimental para el desarrollo del prototipo, la cual consiste en seis etapas: Metodología para la obtención de un sistema embebido confiable basado en el microcontrolador ATmega328P; Metodología para la caracterización y el desarrollo de interfaces capacitivas basadas en sistemas embebidos Metodología para el desarrollo de un sistema de comunicación GSM, Metodología para la obtención de un sistema embebido para el registro de datos inerciales (enfocados en el reconocimiento de actividades humanas), Metodología para la obtención de un modelo de Redes Neuronales para la inferencia en el reconocimiento de actividades humanas y, finalmente, la Metodología para la integración de una herramienta de asistencia basada en un sistema neuromórfico para la detección de actividades humanas, donde se integran los resultados de las cinco etapas

anteriores.

En el capítulo tres se presenta el desarrollo experimental del trabajo, haciendo énfasis en la metodología planteada en el capítulo dos, resaltando detalles que involucran los procesos de manufactura, el desarrollo de software y la integración de los resultados de cada etapa del proyecto.

Finalmente, en el capítulo cuatro se presentan los resultados estructurados por cada etapa del desarrollo del trabajo, realizando un análisis de los gráficos que se presentarán.

CAPÍTULO 1: FUNDAMENTACIÓN Y ANÁLISIS PRELIMINAR PARA EL DESARROLLO DE UN SISTEMA NEUROMÓRFICO ENFOCADO A HERRAMIENTAS PARA RECONOCIMIENTO DE ACTIVIDADES EN ADULTOS MAYORES

En el contexto del envejecimiento poblacional y el incremento en la incidencia de caídas en personas adultas mayores, surge la necesidad de desarrollar soluciones tecnológicas que permitan mejorar su calidad de vida, preservar su autonomía y ofrecer una respuesta oportuna ante situaciones de riesgo. De acuerdo con la Organización Mundial de la Salud, las caídas representan una de las principales causas de lesiones y mortalidad no intencionada, siendo los adultos mayores de 60 años el grupo más vulnerable.

El presente capítulo expone la fundamentación del proyecto de investigación, así como el análisis preliminar que orienta el diseño e implementación de un sistema neuromórfico para el reconocimiento de actividades humanas en una prenda (Wearable). Este sistema busca no solo detectar caídas, sino también reconocer señales inerciales a partir de los patrones cinemáticos de las distintas actividades seleccionadas para su entrenamiento, brindando así un mecanismo de asistencia versátil y adaptable a las necesidades de este sector poblacional.

Se aborda en primer lugar el planteamiento del problema, enmarcando la relevancia social y tecnológica de la propuesta. Posteriormente, se presenta la justificación del proyecto, resaltando su potencial para contribuir a la seguridad y bienestar de los adultos mayores. Asimismo, se formula la hipótesis central y se definen tanto el objetivo general como los objetivos específicos, que guiarán el desarrollo del prototipo, desde el diseño electrónico y la selección de materiales hasta la validación de su funcionalidad y ergonomía.

De esta manera, este capítulo sienta las bases conceptuales y técnicas sobre las que se desarrollará la herramienta de asistencia basada en un sistema neuromórfico, estableciendo los lineamientos que permitirán su implementación como herramienta de asistencia confiable, portátil y centrada en las necesidades reales de la población seleccionada.

1.1 PLANTEAMIENTO DEL PROBLEMA

En 2021, la Organización Mundial de la Salud declaró que las caídas son la segunda causa mundial de muerte por traumatismos involuntarios.

Se calcula que anualmente fallecen 684,000 personas a nivel mundial por causa de caídas, de las cuáles las personas adultas mayores de 60 años presentan la mayor tasa.

La vejez se caracteriza por la aparición de varios estados de salud complejos que se conocen habitualmente por el nombre de síndromes geriátricos. Por lo general, son consecuencia de múltiples factores subyacentes que incluyen, entre otros, la fragilidad, las caídas y la movilidad.

Si bien la problemática del envejecimiento poblacional y las consecuencias de las caídas en adultos mayores han sido ampliamente documentadas, la disponibilidad de soluciones tecnológicas que permiten detectar caídas, reconocer actividades humanas y facilitar la comunicación remota en caso de emergencia es limitada.

Las herramientas existentes suelen abordar funciones específicas, como el monitoreo de variables fisiológicas o la asistencia a la movilidad, pero pocas combinan en un solo sistema características clave, como la portabilidad y la capacidad de procesamiento local de señales para una respuesta inmediata en el reconocimiento de actividades humanas.

Un desafío crítico en el diseño de tecnologías vestibles (wearables) para adultos mayores es lograr un balance entre funcionalidad y portabilidad. Esto implica considerar desde la arquitectura electrónica y la selección de sensores hasta la implementación de herramientas de Inteligencia Artificial que validen el funcionamiento del sistema.

En este contexto, se plantea la necesidad de desarrollar un sistema neuromórfico que integre unidades sensoriales en una prenda y que sea capaz de procesar las señales en tiempo real, así como de comunicarse de forma inalámbrica. Esta propuesta requiere resolver limitaciones técnicas relacionadas con la disposición de los sensores, la manipulación de componentes de montaje superficial, la exploración de algoritmos de Inteligencia Artificial para la creación de modelos de inferencia y la implementación de la IA Embebida, con el fin de obtener una herramienta de asistencia confiable y adaptada a las necesidades reales de los adultos mayores.

1.2 JUSTIFICACIÓN

El envejecimiento poblacional conlleva no solo un aumento en la incidencia de caídas y pérdida de movilidad en adultos mayores, sino también una mayor vulnerabilidad social y emocional. La soledad no deseada, resultado de la pérdida de seres queridos o del distanciamiento de familiares, puede agravar las dificultades asociadas a la dependencia física, psicológica o cognitiva, afectando la calidad de vida y la autonomía personal.

En contraste con las soluciones existentes, este proyecto plantea una integración innovadora de tecnologías en una prenda vestible, donde el diseño electrónico, la ergonomía y la portabilidad se consideran desde la etapa inicial de desarrollo. La incorporación de sensores, procesamiento embebido y comunicación inalámbrica busca no solo resolver limitaciones técnicas presentes en dispositivos comerciales, sino también garantizar una experiencia de uso cómoda y discreta que fomente la aceptación por parte de los adultos mayores.

Este trabajo de investigación se justifica por su contribución al desarrollo de una herramienta de asistencia vestible (E-WEAR) que, mediante sensores y un sistema embebido, ofrece un monitoreo continuo y discreto de las actividades humanas de las personas mayores. Esta solución contribuye a la seguridad, reducir los tiempos de respuesta ante incidentes y brindar tranquilidad tanto al usuario como a sus familiares o cuidadores, abordando así de forma simultánea aspectos técnicos y humanos del cuidado en la vejez.

1.3 HIPÓTESIS

Es posible diseñar, manufacturar e implementar un sistema embebido neuromórfico con la capacidad de inferir el reconocimiento de actividades humanas de manera eficaz, enfocado en satisfacer las necesidades de cuidado de los adultos mayores.

1.4 OBJETIVOS

1.4.1 OBJETIVO GENERAL

Diseñar, manufacturar e implementar un sistema embebido neuromórfico para el reconocimiento de actividades humanas, tales como caminar, levantarse, sentarse y agacharse; y que sirva como herramienta de alerta enfocada a las necesidades de los adultos mayores.

1.4.2 OBJETIVOS ESPECÍFICOS

- Integrar un sistema embebido basado en un microcontrolador para gestionar el procesamiento de señales de los sensores, así como la comunicación inalámbrica y GSM.
- Llevar a cabo el estudio, evaluación y la implementación de diferentes algoritmos de Inteligencia Artificial a fin de crear un modelo de inferencia.
- Integrar las unidades propuestas del sistema embebido en una herramienta de alerta vestible para validar su funcionamiento.

1.5 ESTADO DEL ARTE

1.5.1 INNOVACION EN EL DESARROLLO DE ALGORITMOS DE DEEP LEARNING PARA APLICACIONES BASADAS EN EL RECONOCIMIENTO DE ACTIVIDADES HUMANAS (HAR) MEDIANTE SENSORES INERCIALES

Los sensores de movimiento inercial permiten medir variables como la aceleración y la velocidad angular de un cuerpo; dado que esta unidad sensorial está integrada por un acelerómetro y un giroscopio. El desarrollo de aplicaciones mediante estos sensores ha sido constante debido a la versatilidad y fácil integración en prototipos, tanto para identificar movimientos como para conocer la orientación de un cuerpo.

En [4] se desarrolla una herramienta de detección de caídas basada en un acelerómetro ADXL345, un microcontrolador Raspberry Pi, un zumbador para alertar sobre la caída y una batería con interruptor para alimentar el sistema. En el trabajo se monitorean movimientos como estar de pie, estar acostado, caminar, sentarse, acostarse y también se analizan dos tipos

de caídas: caídas hacia adelante y caídas desde una cama. En el trabajo se remarca que el grupo de usuarios para las pruebas del prototipo estuvo compuesto por 5 hombres y 5 mujeres de entre 20 y 25 años, con una altura de entre 1.50 metros y 1.70 metros.

Algunas de las desventajas del prototipo son el consumo energético del sistema completo, ya que al usar una Raspberry Pi, el consumo se eleva debido a la arquitectura de la tarjeta del microcontrolador, lo que reduce la autonomía en aplicaciones enfocadas en el monitoreo del usuario. También se toma en cuenta la portabilidad, ya que, a pesar de estar encapsulado el sistema, no pretende ser un sistema compacto para poder ser integrado de manera cómoda en el cuerpo de una persona, como lo sería en una prenda de vestir.

Por otra parte, en [5] se desarrolla una herramienta de detección de caídas para personas mayores mediante un acelerómetro con giroscopio y sensores de ubicación en tiempo real (GPS).

Su funcionamiento se basa en detectar la caída y, en ese momento enviar la información a sus familiares desde el número de teléfono almacenado en la programación de la herramienta.

La arquitectura de la herramienta radica en un enclosure impreso en 3D con dimensiones de 3.4x7x3 cm, con la finalidad de que dicho enclosure se introduzca en el bolsillo del pantalón del usuario.

En el trabajo se muestran los resultados de una serie de 10 caídas, de las cuáles 9 notificaron la caída correctamente en Telegram, por lo que su tasa de éxito fue del 90 %.

En el trabajo realizado por Ascioğlu [6], publicado en la revista de IEEE Access en 2020, se realiza una propuesta de prototipos Wearables para el monitoreo de actividades físicas básicas y complejas en la vida cotidiana de los usuarios.

El prototipo consta de 4 sistemas de recolección y transmisión de datos (dispositivos esclavos) y un dispositivo el cual se encarga de procesar los datos y guardarlos en una red creada por el dispositivo, con el propósito de postprocesar los datos una vez recolectados.

En el trabajo se registran datos inerciales en movimientos básicos como caminar, sentarse y levantarse de un asiento, subir y un escalón, correr, saltar, entre otros. Para los movimientos

más complejos se registraron actividades como limpiar una mesa, beber de un vaso, abrir y cerrar un refrigerador, lavarse las manos y escribir en una hoja.

El grupo de estudio fue un total de 60 usuarios, 37 hombres y 23 mujeres dentro de un rango de edad de 20 a 40 años, un rango de altura entre 160 y 195 cm, y un rango de peso entre 50 y 103 kg.

Cada usuario registró un total de 13 actividades, siendo 8 actividades básicas y 5 actividades complejas.

Dentro de las desventajas de este proyecto cabe señalar que la inferencia de los resultados se hace de manera remota postprocesando los datos en modelos generados una vez realizada la captura. Por otra parte, la modularidad de los dispositivos esclavos hace que la aplicación dependa de diversas fuentes de energía (una por dispositivo de adquisición de datos), además de hacer la aplicación wearable un poco incómoda para el usuario.

Sin embargo, la propuesta es buena para el propósito que se presenta en el trabajo, siendo el objetivo evaluar los diferentes modelos y técnicas de Deep learning para la clasificación de movimientos inerciales. Además, dado este contexto, el tener diferentes dispositivos de adquisición de datos colocados en diferentes partes del cuerpo del usuario permite un registro más preciso del movimiento del usuario.

Una aportación interesante al mercado de las herramientas de monitoreo son los Wearables en forma de anillo, como el desarrollado por Brian Coffen en 2021, presentado en la Conferencia Internacional de E-Health Networking, Application and Services de la IEEE.[7]

En el trabajo se evalúan distintas técnicas y algoritmos de Deep learning para la identificación de gestos mediante el anillo mencionado anteriormente. La arquitectura del Wearable consiste en un acelerómetro y un módulo de comunicación Bluetooth integrado para comunicación UART.

Los datos registrados se postprocesan mediante algoritmos diseñados en Python para guardarlos en archivos CSV, registrando aproximadamente 2 segundos del movimiento por medición.

Para el diseño de los modelos de Deep learning principalmente se utilizaron librerías como

TensorFlow y Keras.

Durante el desarrollo del trabajo se evalúan en total 10 gestos diferentes mediante el acelerómetro que incluye el prototipo del anillo inteligente. Estos gestos caracterizados por un patrón específico de movimiento del dedo índice derecho.

En [8] se presenta el desarrollo de diferentes modelos de Deep Learning, en el cuál se evalúan movimientos de actividades básicas como caminar o subir escalones mediante las mediciones registradas por el acelerómetro y giroscopio de un teléfono celular de modelo Samsung Galaxy S II.

En el trabajo, el grupo de estudio consta de 30 personas dentro de un rango de 19 a 48 años. Cada uno fue evaluado en realizar 6 actividades cotidianas como lo son: Estar de pie, Sentarse, Acostarse, Caminar, Subir escalones y Bajar escalones.

En el trabajo se evalúan los resultados aplicando inferencia mediante un modelo de CNN (Convolutional Neural Networks) o Redes Neuronales Convolucionales aplicando LSTM (Long Short-Term Memory).

LSTM se especializa en modelar dependencias temporales o secuenciales. Es muy útil cuando los datos tienen una dimensión de tiempo, como series temporales o secuencias.

Una de las desventajas es que el dispositivo de registro de datos inerciales, aunque es un dispositivo que hoy en día cualquier persona lo tiene, tal vez no con las mismas funciones, puede no ser tan práctico colocar el teléfono celular en una zona estratégica para monitorear con precisión el movimiento inercial del usuario. Por otra parte se hace más complicada la configuración para poder realizar una inferencia en tiempo real.

Por otra parte en el trabajo presentado en la revista Sensors de MDPI en 2023 por Shakerian [9], se muestra el desarrollo de un prototipo para monitorear en tiempo real actividades de la vida cotidiana.

El prototipo consiste en un acelerómetro con giroscopio de bajo costo, así como un micro-controlador con soporte para comunicación Wi-Fi para transmitir los datos a un servidor y posteriormente enviarlos a una aplicación móvil.

El dispositivo de adquisición de datos fue adaptado a un cinturón de velcro mediante piezas impresas en 3D, siendo la "hebilla" del cinturón el dispositivo de monitoreo inercial.

En el trabajo se aplican y se evalúan modelos de Redes Neuronales Convolucionales junto con técnicas de LSTM (Long Short-Term Memory) o Memoria a Corto-Largo plazo.

Una de las ventajas de la arquitectura presentada en este trabajo es la posibilidad de enviar los datos a un dispositivo celular mediante el servidor generado por el segundo microcontrolador.

Una desventaja es el requerimiento de diferentes módulos para poder llevar a cabo el monitoreo en tiempo real y posiblemente el costo del hardware que conforma el sistema.

Basándonos en la tendencia de utilizar los dispositivos celulares como medio para registrar los datos inerciales, en [10] se presenta un trabajo en donde se evalúan 18 actividades físicas, incluyendo movimientos del cuerpo y de las manos, así como movimientos en acciones como comer, utilizando los sensores integrados en smartphones y smartwatches.

Algunas de las actividades a registrar fueron actividades como caminar, correr, subir escaleras, sentarse, patear una pelota, botar con la pelota de baloncesto, cachar una pelota de tenis, escribir, lavarse los dientes, guardar ropa, entre otros.

Con el objetivo de extraer las características más importantes de los datos inerciales, se recurrió al diseño de un modelo de Redes Neuronales Convolucionales con un conjunto de Redes Neuronales Recurrentes.

Para la clasificación de las actividades se utilizó MPNN.

Una MPNN es una arquitectura de Graph Neural Network (GNN) que modela la interacción entre nodos de un grafo a través de un proceso de “paso de mensajes” (message passing). Es especialmente útil cuando los datos tienen una estructura no euclidiana, como moléculas, redes sociales, mapas de tráfico o circuitos eléctricos.

Una de las ventajas presentadas en este trabajo es la comodidad del registro de datos, desde uno de los dispositivos más comunes hoy en día, como lo es el teléfono celular, así como un smartwatch, que hoy en día es una herramienta muy utilizada en sectores deportivos, de salud y de monitoreo.

Por otra parte en [11] se propone una arquitectura basada en sensores inerciales de smartphones para el reconocimiento de actividades humanas (HAR, por sus siglas en inglés) en tiempo real, empleando modelos de aprendizaje profundo. Aprovechando sensores integrados como acelerómetros y giroscopios, los autores recopilan datos de movimiento y aplican técnicas de preprocesamiento (eliminación de ruido, normalización y segmentación) para extraer vectores de características relevantes.

Se exploran y comparan cinco modelos de clasificación: Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), Bidirectional LSTM (BLSTM), Multilayer Perceptron (MLP) y Support Vector Machines (SVM), utilizando dos conjuntos de datos públicos ampliamente utilizados: UCI HAR y Pamap2. La metodología incluye miles de experimentos con diversas métricas (precisión, recall, F1-score y exactitud), demostrando que la red CNN ofrece el mejor rendimiento general, seguida por BLSTM.

Entre las contribuciones que aporta este trabajo destaca el uso de una arquitectura CNN para extraer características espaciales-temporales relevantes del movimiento humano, así como la evaluación de los modelos en dos datasets con diferente complejidad y número de clases.

Los resultados muestran que las CNNs son efectivas para identificar actividades cotidianas como caminar, correr o estar sentado, incluso sin necesidad de un diseño manual de características, lo que las hace ideales para implementaciones en tiempo real en dispositivos móviles. Se concluye que los smartphones pueden ser una herramienta poderosa y práctica para aplicaciones de monitoreo inteligente en salud, hogares inteligentes y asistencia personal.

Continuando con los avances en el análisis de actividades inerciales mediante sensores integrados en smartphones, En [12] se desarrolla un sistema de reconocimiento de actividades humanas (HAR, por sus siglas en inglés) empleando únicamente acelerómetros integrados en teléfonos celulares, sin requerir sensores especializados ni configuraciones complejas. El estudio utilizó el conjunto de datos WISDM, el cual recopila información de aceleración tridimensional (ejes x, y, z) de 36 participantes realizando seis actividades comunes: caminar, correr, estar sentado, de pie, subir escaleras y bajarlas. Para la clasificación de dichas actividades, los autores implementaron un modelo de red neuronal convolucional bidimensional (2D

CNN), con un enfoque centrado en ventanas de 80 muestras y datos normalizados para reducir varianza y sobreajuste.

El modelo alcanzó una precisión promedio del 89.67 %, destacándose en la identificación de actividades estáticas como estar sentado y de pie (ambas con 100 % de aciertos), así como en actividades dinámicas como correr (94 %) y caminar (94 %). Sin embargo, presentó confusión entre actividades de patrones similares, como subir y bajar escaleras, siendo esta última reconocida con un 61 % de precisión. A pesar de no superar en promedio a un modelo LSTM comparado en el mismo estudio (92.33 % de precisión), la propuesta basada en CNN mostró ventajas en simplicidad, menor requerimiento computacional y un buen desempeño general. Esta investigación demuestra la viabilidad de utilizar acelerómetros de smartphones y modelos ligeros de aprendizaje profundo para aplicaciones de HAR, siendo una opción adecuada para sistemas embebidos o móviles con recursos limitados.

Un trabajo destacable en este ámbito es el propuesto por Alessandrini en 2021 [13] donde se desarrolló un sistema de reconocimiento de actividades humanas (HAR) que emplea una red neuronal recurrente (RNN) basada en celdas LSTM, diseñada para operar en sistemas embebidos de bajo costo y bajo consumo, utilizando como entrada señales de fotopletismografía (PPG) y acelerometría triaxial obtenidas desde la muñeca. A diferencia de muchas propuestas previas que utilizan únicamente sensores inerciales, este trabajo explora el aprovechamiento del sensor PPG —común en dispositivos wearables— como fuente adicional para mejorar el desempeño del sistema HAR, incluso ante artefactos de movimiento.

El sistema fue entrenado con un conjunto de datos público, que incluye señales de PPG y acelerómetro de siete sujetos adultos realizando tres actividades: descanso, sentadillas y uso de stepper. Los datos fueron capturados con una plataforma MAXREFDES100 a 400 Hz. Para el entrenamiento se usaron los datos de cinco sujetos y los de los otros dos se reservaron para pruebas. Se implementó una estrategia de preprocesamiento que incluye normalización por ventana, eliminación de artefactos, eliminación del offset gravitacional en acelerómetros y reducción del tamaño de entrada mediante decimación sin interpolación, para facilitar la implementación en sistemas embebidos.

El artículo destaca que la precisión del modelo embebido es idéntica a la obtenida en PC, demostrando que no fue necesario aplicar técnicas de cuantización o compresión adicionales. Además, con solo 25,283 parámetros entrenables, el sistema se muestra como una alternativa viable para aplicaciones portátiles autónomas en tiempo real. Finalmente, se compara el rendimiento de su propuesta con trabajos previos, resaltando que pocos han logrado implementar redes profundas con señales PPG directamente en microcontroladores.

Un paso importante en la metodología para el desarrollo de aplicaciones capaces de clasificar datasets o clases con datos provenientes de sensores inerciales es el preprocesamiento, el acondicionamiento y la manipulación de los datos.

Al hablar de movimientos inerciales, un movimiento puede tener hasta 100 muestras en un lapso de un par de segundos, por lo que seleccionar las características más importantes de cada movimiento puede llegar a ser un reto, principalmente si se pretende implementar en un sistema con recursos limitados, como lo es un sistema embebido.

En [14] se realizó un estudio exhaustivo orientado a identificar las características más relevantes para el reconocimiento de actividades humanas (HAR) utilizando acelerómetros triaxiales, con el objetivo de optimizar el rendimiento de los clasificadores y reducir la carga computacional en sistemas de cómputo de recursos limitados. Para ello, se extrajeron 193 características (muchas de ellas inéditas en este campo) de señales de aceleración registradas en cuatro conjuntos de datos públicos: WISDM, HARDS (HHAR), SBRHA y PAMAP2. Estos datasets difieren en frecuencia de muestreo, ubicación de sensores y complejidad de actividades, lo cual permitió analizar de manera integral la influencia de estas variables en la relevancia de las características.

El enfoque metodológico incluyó segmentación con ventanas deslizantes de 2.5, extracción de características en dominios temporal y frecuencial, y una normalización y discretización para evitar sesgo de datos.

Los resultados muestran que características simples como la media, desviación estándar y RMS fueron altamente informativas para datasets con actividades simples y baja frecuencia de muestreo, mientras que en datasets con mayor complejidad y frecuencia, las características

relacionadas con regularidad, recurrencia y caos ganaron relevancia. Además, se concluyó que la ubicación del sensor no tiene un efecto significativo en la selección de características, lo cual sugiere flexibilidad en aplicaciones portables.

En términos de clasificación, los modelos SVM y KNN lograron hasta 93.24 % de precisión con el dataset WISDM usando solo 30 características significativas, superando el rendimiento reportado en estudios previos con ese mismo conjunto de datos. Esto respalda que una selección adecuada de características puede superar incluso a modelos más complejos o con mayor número de entradas.

Este trabajo es especialmente relevante para el diseño de sistemas embebidos eficientes, pues permite optimizar el uso de recursos sin sacrificar precisión, aportando una base sólida para la reducción de dimensionalidad en el Reconocimiento de Actividades Humanas y su implementación en tiempo real.

En [15] se presenta una revisión exhaustiva de los avances recientes en el reconocimiento de actividades humanas (HAR, por sus siglas en inglés) mediante sensores portables y técnicas de aprendizaje profundo (DL). Su trabajo clasifica y analiza 176 artículos seleccionados con base en relevancia técnica y novedad, excluyendo aquellos que utilizan imágenes, sensores ambientales o que no aplican deep learning. La revisión abarca tanto aspectos teóricos como prácticos, ofreciendo un marco general para investigadores nuevos y experimentados en el área.

El artículo organiza los estudios en función de cuatro ejes principales:

Tipo de sensor: sensores físicos (IMU, GPS, piezoeléctricos) y fisiológicos (ECG, EMG, PPG, etc.).

Aplicación: salud, estilo de vida, rehabilitación, interacción humano-computadora, entre otros.

Arquitecturas de aprendizaje profundo: incluyendo Autoencoders (AE), Deep Belief Networks (DBN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN y LSTM), Generative Adversarial Networks (GAN) y Deep Reinforcement Learning (DRL).

La revisión de Zhang et al. (2022) constituye una de las síntesis más completas y actuales en

el campo del reconocimiento de actividades humanas (HAR) mediante sensores modulares y aprendizaje profundo. En ella se identifican las arquitecturas de red más utilizadas (CNN, RNN/LSTM, AE, GAN, DRL), así como los principales desafíos que enfrenta la comunidad, como la variabilidad en la ubicación de sensores, el despliegue eficiente en hardware embebido, la generalización de modelos y la privacidad de los datos.

Uno de los puntos clave del artículo es la necesidad de adaptar modelos de aprendizaje profundo a sistemas embebidos con restricciones de procesamiento y energía, resaltando que, aunque las redes convolucionales y recurrentes ofrecen alta precisión, su implementación directa en microcontroladores requiere técnicas de optimización como cuantización, pruning o arquitecturas modulares.

En contraste, la propuesta desarrollada en esta tesis busca precisamente atender dicho reto, implementando un sistema HAR embebido que:

Hace uso de Unidades de Medición Inercial (IMUs) con acelerómetro y giroscopio para inferir patrones de movimiento.

Implemente una red neuronal entrenada previamente (en Keras) e inferida en tiempo real dentro del Portenta H7, una plataforma que combina capacidad de cómputo suficiente (Cortex-M7 + Cortex-M4) con bajo consumo energético.

Se enfoca específicamente en la detección de caídas, una subcategoría crítica dentro del HAR para adultos mayores.

En el desarrollo de trabajos de Reconocimiento de Actividades Humanas como los analizados, es de suma importancia poner a consideración el equipamiento del usuario, ya que, como se mencionó antes, factores como el peso o incluso la estatura del usuario pueden ser características que marquen la medición o el registro del movimiento.

Este enfoque de análisis se presentó en [16] donde investigaron cómo el equipamiento especializado y la carga adicional afectan la precisión de los sistemas de reconocimiento de actividades humanas (HAR) basados en acelerómetros, específicamente en ambientes extremos como el montañismo. A diferencia de los estudios tradicionales en ambientes controlados, este

trabajo explora el impacto que tienen factores como el uso de botas rígidas de alpinismo y mochilas con carga (12 kg) en la clasificación de actividades.

Para ello, se equipó a ocho participantes con un traje de captura de movimiento Perception Neuron, que integra 17 acelerómetros distribuidos en todo el cuerpo, muestreados a 120 Hz. Se realizaron siete actividades simuladas de montañismo en dos condiciones: sin equipo especializado y con equipo completo. Se extrajeron 41 características estadísticas y frecuenciales de las señales y se evaluaron cinco algoritmos de clasificación: SVM, árboles de decisión, k-NN, k-NN ponderado y bosques aleatorios.

Entre los hallazgos, el SVM obtuvo la mejor precisión, alcanzando hasta 93.36 % de precisión con un solo acelerómetro en el pie derecho, y hasta 96.29 % combinando solo dos sensores (cadera y tibia derecha). Sin embargo, al entrenar modelos con datos sin equipamiento y probar con datos de usuarios equipados, la precisión cayó drásticamente, con una media de solo 51.79 %, evidenciando que el uso de equipo altera significativamente los patrones de aceleración.

El estudio también determinó que la precisión mejora al aumentar el número de sensores, pero con disminución de retornos después de dos sensores, haciendo impráctico el uso de muchos sensores en ambientes extremos por la complejidad, el consumo energético y el impacto en la movilidad del usuario.

En [17] propusieron un modelo híbrido de aprendizaje profundo denominado 1D-CNN-BiLSTM, diseñado para el reconocimiento de actividades humanas (HAR) a partir de datos de sensores portables. El modelo combina la capacidad de las redes convolucionales unidimensionales (1D-CNN) para extraer características discriminantes de series temporales, con la habilidad de las redes Bidirectional Long Short-Term Memory (BiLSTM) para capturar dependencias temporales en ambas direcciones del tiempo. Esto permite una representación más completa de las transiciones de actividad.

El sistema fue evaluado en tres datasets públicos ampliamente utilizados en HAR:

UCI-HAR: acelerómetro y giroscopio en la cintura, seis actividades básicas (95.48 % de precisión).

Motion Sense: múltiples variables inerciales en smartphones, seis actividades dinámicas (94.17 % de precisión).

Single Accelerometer: aceleración registrada en el pecho, cinco actividades seleccionadas (100 % de precisión).

El diseño del modelo incluye capas convolucionales con activación ReLU, capas de dropout para evitar sobreajuste, max-pooling y finalmente, BiLSTM con fusión por concatenación de los estados forward y backward. Además, se realizó una optimización de hiperparámetros por grid search, determinando que el optimizador SGD con momentum, un tamaño de lote de 64 y la fusión por concatenación proporcionaban el mejor desempeño.

Por otra parte, diversos estudios han señalado que los métodos tradicionales de segmentación de series temporales, basados en ventanas estáticas de tiempo, pueden limitar la capacidad de los modelos para capturar adecuadamente los patrones cinemáticos de ciertas actividades. Una ventana muy amplia puede incluir información irrelevante, mientras que una demasiado corta puede omitir partes críticas de movimientos prolongados. [15]

Las tecnologías portátiles inteligentes impulsadas por inteligencia artificial no solo tienen aplicaciones en el ámbito médico, sino también en la gestión ambiental y en el desarrollo de entornos urbanos inteligentes.[18]

En síntesis, los trabajos analizados reflejan el rápido avance en el uso de sensores inerciales y algoritmos de aprendizaje profundo para el Reconocimiento de Actividades Humanas (HAR), evidenciando una tendencia hacia sistemas cada vez más compactos, precisos y orientados a la implementación en tiempo real. Sin embargo, persisten desafíos importantes relacionados con la portabilidad, el consumo energético y la capacidad de ejecutar inferencia local en plataformas embebidas de recursos limitados. A partir de esta revisión, se identifica una oportunidad de innovación en el desarrollo de sistemas HAR que combinen eficiencia computacional, autonomía energética y precisión en la clasificación, particularmente en aplicaciones críticas como la detección de caídas en adultos mayores. En este contexto, la presente tesis busca contribuir al estado del arte mediante el diseño de un sistema neuromórfico embebido capaz de realizar inferencia en tiempo real, manteniendo un equilibrio entre rendimiento, consumo

energético y viabilidad práctica para el monitoreo continuo de la actividad humana.

A partir del análisis del estado del arte, se observa que el reconocimiento de actividades humanas mediante sensores inerciales constituye un campo multidisciplinario que integra conocimientos de electrónica, procesamiento de señales y aprendizaje profundo. Para sustentar el desarrollo del sistema propuesto en esta tesis, es necesario establecer las bases teóricas que fundamentan su diseño e implementación. En la siguiente sección se abordan los principios de funcionamiento de los sensores inerciales, las técnicas de adquisición y procesamiento de datos, así como los conceptos esenciales de las redes neuronales profundas aplicadas al reconocimiento de patrones de movimiento, proporcionando el marco conceptual que permitirá comprender la arquitectura y operación del sistema neuromórfico desarrollado.

1.6 REVISIÓN TEÓRICA DE LOS SISTEMAS UTILIZADOS PARA EL DESARROLLO DE HERRAMIENTAS DE ASISTENCIA EN ADULTOS MAYORES

El presente marco teórico tiene como objetivo establecer las bases conceptuales y científicas que sustentan esta investigación, proporcionando un contexto sólido para el desarrollo de un sistema neuromórfico basado en sensores capacitivos aplicados a herramientas de asistencia para adultos mayores.

En primer lugar, se presenta una revisión teórica de los sistemas utilizados en el desarrollo de tecnologías de asistencia dirigidas a la población adulta mayor (1.4). En esta sección, se analizan aspectos relacionados con la ergonomía y la portabilidad de los sistemas wearables, así como un estudio comparativo de sus capacidades de comunicación y de los microcontroladores empleados para el procesamiento automático de datos. Dicho análisis incluye la inferencia de categorías de actividad y el procesamiento de información proveniente de sensores, lo cual resulta fundamental para la interpretación del movimiento humano en aplicaciones de asistencia.

Posteriormente, se abordan los aspectos generales para el desarrollo de un sistema neuromórfico (1.5), donde se exponen los fundamentos teóricos y conceptuales que permiten comprender los principios de funcionamiento de este tipo de sistemas. En esta parte, se incluyen los ele-

mentos técnicos necesarios para su diseño e implementación, considerando tanto los principios físicos como computacionales involucrados, así como su integración en entornos prácticos de uso.

Finalmente, se establece la relación de los temas abordados con el enfoque de la investigación, justificando la pertinencia de los sistemas neuromórficos basados en unidades de medición inercial (IMU's) como una solución tecnológica eficiente y viable para responder a las necesidades de asistencia de los adultos mayores. 1.4 Revisión teórica de los sistemas utilizados para el desarrollo de herramientas de asistencia en adultos mayores.

El envejecimiento de la población a nivel mundial es un fenómeno que ha cobrado cada vez más relevancia en los últimos años. De acuerdo con las estadísticas más recientes publicadas por la Organización Mundial de la Salud (OMS, 2024), se proyecta que entre 2015 y 2050 el porcentaje de personas mayores de 60 años prácticamente se duplicará, incrementándose del 12 % al 22 % de la población global.

En la actualidad, la mayoría de los países enfrentan desafíos significativos para adaptar sus sistemas de salud y asistencia social con el fin de responder al importante cambio demográfico que representa el aumento de la población adulta mayor. Para el año 2020, el número de personas de 60 años o más superó al de niños menores de cinco años. Asimismo, se estima que hacia 2050, alrededor del 80 % de los adultos mayores residirá en países con ingresos bajos o medios [19]. Desde una perspectiva biológica, el envejecimiento se entiende como un proceso natural e inevitable que resulta de la acumulación progresiva de daños a nivel celular y molecular. Estos cambios ocasionan una disminución paulatina de las capacidades físicas, lo que hace a este grupo poblacional más vulnerable a diversas condiciones, entre ellas los síndromes geriátricos. Los síndromes geriátricos son un conjunto de problemas comunes en adultos mayores que no se clasifican fácilmente dentro de enfermedades específicas, pero que tienen un impacto considerable en su calidad de vida y en su nivel de funcionalidad. Entre estos se encuentran aspectos como:

- Inmovilidad
- Incontinencia

- Deterioro Cognitivo
- Delirium
- Fragilidad
- Caídas

Las caídas son una de las causas principales de lesiones y muerte en personas mayores, y su aparición suele estar relacionada con múltiples factores, incluyendo pérdida de fuerza, alteraciones del equilibrio, problemas visuales, uso de medicamentos y condiciones del entorno.

Las caídas no solo representan un evento físico, sino también un marcador clínico de vulnerabilidad que puede indicar deterioro funcional y riesgo de dependencia de un cuidador.

Es por ello que se han desarrollado herramientas de asistencia de diferentes gamas enfocadas a la población de la tercera edad con el objetivo de cubrir de manera parcial o completa dichas necesidades. A continuación, se presenta una breve descripción conceptual de las herramientas tecnológicas de asistencia.

La tecnología de asistencia se refiere a una amplia gama de dispositivos y servicios diseñados para mejorar la independencia y calidad de vida de las personas con algún tipo de discapacidad, ya sea física, cognitiva o sensorial [20]. Entre los beneficios que aporta dicha tecnología destacan una mayor autonomía para realizar actividades cotidianas, una mejor comunicación y una mayor accesibilidad.

Generalmente, la tecnología de asistencia se clasifica en tres tipos:

- Tecnología Baja
- Tecnología Media
- Tecnología Alta

La tecnología baja hace referencia a herramientas sencillas, como agarraderas para lápices, reglas u otros elementos que facilitan el acceso a los objetos necesarios. Por otro lado, está la

tecnología media, que incluye dispositivos como lupas, materiales en braille o asientos adaptables al usuario, mientras que la alta tecnología comprende los dispositivos más avanzados, como lectores de pantalla, convertidores de texto a voz y sistemas de monitoreo, entre otros.

En este último nivel se encuentran herramientas que requieren un funcionamiento autónomo, adaptable y eficiente, características que son posibles gracias a la implementación de sistemas embebidos. Estos son dispositivos electrónicos diseñados para realizar tareas específicas sin necesidad de una interacción continua con el usuario. Debido a la variedad de aplicaciones y tecnologías, los sistemas embebidos presentan diferentes formas y tamaños, pero suelen caracterizarse por ser pequeños y contar con recursos limitados.

Con el continuo avance en la industria de los semiconductores, los sistemas embebidos han evolucionado hacia arquitecturas cada vez más compactas y optimizadas. En esencia, un sistema embebido es un sistema computacional diseñado para ejecutar una función específica o un conjunto limitado de funciones dentro de un sistema mayor. A diferencia de los sistemas de propósito general, como una computadora personal, los sistemas embebidos están optimizados para tareas concretas, lo que los hace más eficientes, de bajo consumo energético y, en muchos casos, capaces de operar en tiempo real [21].

La alta tecnología actualmente sigue en auge por el avance en el desarrollo de herramientas de asistencia confiables, portátiles y ergonómicas que sean de fácil uso para la población de la tercera edad, por lo que dentro de este ámbito los sistemas wearables son los que destacan.

Los sistemas wearables y las aplicaciones mHealth se están convirtiendo en una parte importante de la práctica asistencial, al permitir el registro y la monitorización continua y longitudinal de los parámetros de la salud fuera de las consultas.

Los sistemas portátiles de monitoreo, comúnmente conocidos como wearables, están transformando la biomedicina moderna mediante el uso de tecnologías móviles y digitales que posibilitan la observación continua y prolongada de parámetros de salud sin necesidad de permanecer en un entorno hospitalario.[22]

El desarrollo acelerado del Internet de las Cosas (IoT) ha impulsado la creación de dispositivos electrónicos de tamaño reducido con capacidad de procesamiento, los cuales pueden

incorporarse directamente en el cuerpo humano para diversas aplicaciones biomédicas y de monitoreo.[23]

El monitoreo a largo plazo de parámetros relacionados con la salud y los estilos de vida permite la detección temprana de factores de riesgo, así como el diagnóstico oportuno de enfermedades crónicas o eventos inesperados.

El acceso a información detallada sobre el estado de salud del usuario facilita la implementación de intervenciones personalizadas, fomentando el empoderamiento del paciente y reduciendo el riesgo de enfermedades o situaciones que comprometan su vida.

Este proceso es posible gracias a sensores que recopilan y transmiten datos de manera continua, los cuales se clasifican principalmente en tres categorías: sensores de movimiento, que convierten el movimiento mecánico en señales eléctricas; sensores fisiológicos, que emplean principios ópticos, eléctricos, acústicos o térmicos para medir parámetros vitales como frecuencia cardíaca, temperatura, presión arterial, saturación de oxígeno, así como actividad bioeléctrica (electrocardiografía y electroencefalografía); y sensores bioquímicos, destinados a la medición de sustancias químicas como glucosa o electrolitos. Asimismo, pueden clasificarse según su aplicación: en salud del consumidor, investigación o atención clínica. [24]

1.6.1 ERGONOMÍA Y PORTABILIDAD DE LOS SISTEMAS WEARABLES

El término Wearable se refiere a aquel dispositivo tecnológico que se lleva puesto, es decir, que es portable, como los relojes inteligentes o pulseras, que usualmente están diseñados para analizar y ayudarnos a mejorar nuestra salud. Entre otras cosas, controlan la actividad física, la frecuencia cardíaca, entre otros. En general, proporcionando un aumento en la calidad de vida del usuario. [25]

Uno de los beneficios más representativos de las herramientas wearables es el monitoreo continuo de variables físicas que permiten el paso a aplicaciones médicas o clínicas. Estos dispositivos, además de tener la capacidad de registrar actividades físicas, también pueden monitorear datos vitales como frecuencia cardíaca, patrones de sueño e incluso cumplir funciones como oxímetro. Dicha información puede ser de vital importancia para los profesionales

médicos y permite tomar medidas preventivas para mejorar la salud y el bienestar del usuario.

Como se revisó en la sección anterior, en el estado del arte, actualmente existen diversos avances de este tipo de herramientas, intentando hacer un dispositivo compacto, funcional y portátil para mejorar la calidad de vida del adulto mayor, siendo el wearable más común el reloj inteligente, que por sus características puede llegar a ser una de las mejores soluciones en este ámbito. Sin embargo, es una herramienta que suele ser costosa debido a todas las funciones que se integran en un mismo dispositivo.

Es por ello que en el ámbito científico se buscan soluciones más accesibles a toda la población en la que se integren en parte las mismas funciones, involucrando ahora técnicas revolucionarias como lo es la categorización por algoritmos de Machine Learning o Deep Learning en sistemas embebidos que tengan la función de reconocimiento de actividades humanas, con la posibilidad de establecer un medio de comunicación para notificar un evento anómalo.

1.6.2 ESTUDIO COMPARATIVO DE LAS UNIDADES DE COMUNICACIÓN EN LOS SISTEMAS WEARABLES Y SUS CAPACIDADES

En el desarrollo de sistemas wearables, la capacidad de comunicación es un factor esencial, pues permite la transmisión de datos hacia otros dispositivos o servicios en la nube. Dependiendo del tipo de aplicación, pueden emplearse diferentes tecnologías inalámbricas: Wi-Fi, que ofrece altas velocidades en distancias cortas; Bluetooth, que proporciona bajo consumo energético para comunicación con smartphones u otros dispositivos personales; y GSM/4G, que asegura cobertura a gran escala mediante redes celulares, permitiendo que el wearable funcione de forma autónoma.

En esta subsección se presenta un análisis comparativo de distintos módulos de comunicación externos comúnmente empleados con microcontroladores como el ATmega328P, STM32 o Arduino Portenta H7. El objetivo es identificar las ventajas y limitaciones de cada tecnología en términos de alcance, consumo energético, velocidad de transmisión y costo, de modo que sirva como referencia para seleccionar la opción más adecuada en el diseño de herramientas de asistencia para adultos mayores.

Tabla 1.1: Comparativa de módulos de comunicación utilizados en sistemas embebidos.

Módulo	Tecnología	Velocidad / Alcance	Consumo / Costo	Funciones principales
ESP-01	WI-FI (802.11 b/g/n)	Hasta 72 Mbps, 30–50 m interiores	Moderado–alto / Bajo	Transmisión y carga de datos para monitoreo remoto
HC-05	Bluetooth Clásico (2.0)	3 Mbps, 10–30 m típicos	Bajo / Bajo	Comunicación serie punto a punto para transmisión local
HM-10	Bluetooth Low Energy (BLE 4.0)	1 Mbps, 10–30 m típicos	Muy bajo / Bajo	Comunicación serie punto a punto para transmisión local
RN42	Bluetooth Clásico (2.1)	3 Mbps, 10–100 m (con antena externa)	Bajo / Medio	Comunicación serie punto a punto para transmisión local
SIM800L	GSM/GPRS (2G)	Hasta 85.6 kbps, cobertura celular (km)	Moderado / Medio	Llamadas de voz, SMS, datos GPRS
SIM900A	GSM/GPRS (2G)	Hasta 85.6 kbps, cobertura celular (km)	Moderado / Medio–alto	Llamadas de voz, SMS, datos GPRS
SIM7600	4G LTE Cat-1	10 Mbps, cobertura celular (km)	Alto / Alto	Llamadas de voz, SMS, datos LTE

El análisis comparativo de los módulos de comunicación evidencia que la selección de la tecnología depende directamente de los requerimientos de la aplicación en los sistemas wearables. Los módulos Bluetooth (HC-05, HM-10 y RN42) destacan por su bajo consumo energético y su facilidad de integración con microcontroladores de uso común. En particular, el RN42 resulta una opción adecuada para el registro de datos en la etapa de entrenamiento

de la red neuronal, ya que ofrece una conexión estable bajo el perfil serie (SPP) con un alcance suficiente y un consumo energético reducido, lo que facilita la transmisión confiable de información hacia una computadora o dispositivo móvil sin comprometer la autonomía del prototipo.

Por otra parte, los módulos GSM/GPRS (SIM800L y SIM900A) y el SIM7600/A7670 orientado a 4G LTE, permiten extender la funcionalidad de los sistemas wearables hacia una operación autónoma, sin necesidad de depender de un dispositivo intermedio como un smartphone. En este caso, el SIM800L se considera idóneo para la herramienta de asistencia propuesta en la tesis, ya que integra funciones de llamadas de voz y envío de mensajes SMS, con un consumo energético moderado y un costo accesible en comparación con soluciones más avanzadas como el SIM7600.

Estas características lo convierten en una alternativa balanceada entre funcionalidad y eficiencia, particularmente en aplicaciones dirigidas a adultos mayores, donde la prioridad es la fiabilidad de la comunicación en situaciones de emergencia más que la velocidad de transmisión de datos.

En conclusión, mientras el RN42 se justifica como el módulo más adecuado para la fase de captura de datos y entrenamiento del modelo de reconocimiento de actividades, el SIM800L se establece como la elección más conveniente para la implementación final de la herramienta de asistencia, garantizando conectividad celular, autonomía y facilidad de uso en un entorno real.

Estos módulos necesitan de un microcontrolador para realizar sus funciones, ya que se necesitan una serie de comandos para configurar ambos módulos, con los denominados Comandos AT, mostrados en el ANEXO A del presente documento.

1.6.3 ANÁLISIS COMPARATIVO DE LOS SISTEMAS MICROCONTROLADORES PARA EL PROCESAMIENTO AUTOMÁTICO DE DATOS EN APLICACIONES WEARABLES

El desarrollo de sistemas wearables orientados a la asistencia requiere de microcontroladores con capacidades diferenciadas, que van desde la adquisición y procesamiento primario de

datos de sensores hasta la ejecución de algoritmos de inferencia en tiempo real. La elección del microcontrolador es, por tanto, un aspecto crítico que influye directamente en la autonomía energética, la viabilidad de implementación de modelos de aprendizaje automático y la interacción con módulos de comunicación.

En esta subsección se presenta un análisis comparativo de microcontroladores comúnmente empleados en aplicaciones embebidas: el ATmega328P, ampliamente utilizado en el ecosistema Arduino; el ESP32, reconocido por integrar conectividad inalámbrica y capacidades de procesamiento superiores; el STM32, que ofrece alto rendimiento en aplicaciones de cómputo intensivo; y el Arduino Portenta H7, como representante de una arquitectura de altas prestaciones orientada a la inferencia en inteligencia artificial embebida. El objetivo de esta comparación es establecer criterios de selección basados en el equilibrio entre potencia de procesamiento, consumo energético y adaptabilidad a entornos portátiles, de modo que se sustente la elección final de microcontroladores para el sistema propuesto.

Tabla 1.2: Comparativa de microcontroladores utilizados en sistemas embebidos.

Microcontrolador	Arquitectura / Frecuencia	Memoria (Flash / RAM)	Consumo / Costo	Capacidades principales
ATMega328P	AVR 8 bits, 16 MHz	32 KB / 2 KB	Muy bajo / Bajo	Adecuado para aplicaciones portátiles de registro de datos y sensores.
ESP32	Xtensa LX6 dual-core, 240 MHz	520 KB SRAM + Flash externa	Bajo–moderado / Medio	Ideal para comunicación inalámbrica directa y procesamiento básico.
STM32F4xx	ARM Cortex-M4 (1 núcleo con FPU), 168 MHz	1 MB / 192 KB	Moderado / Medio	Adecuado para procesamiento automático de datos e inferencia de modelos sencillos.
STM32H747 (Portenta H7)	ARM Cortex-M7/M4 dual-core, 480 MHz (M7) / 240 MHz (M4)	2 MB Flash / 1 MB RAM + 16 MB Flash QSPI	Moderado–alto / Alto	Ideal para procesamiento intensivo de datos, IA embebida y tareas en tiempo real.

El análisis comparativo de microcontroladores evidencia que la selección depende directamente del tipo de procesamiento requerido y de la cantidad y resolución de los sensores a conectar en sistemas wearables. El ATmega328P, con su arquitectura AVR de 8 bits, bajo consumo y disponibilidad de entradas analógicas y digitales suficientes, se muestra ideal para el registro de datos de sensores en la fase de construcción de la base de datos para entrenamiento. Su simplicidad y eficiencia energética permiten recopilar información confiable sin incrementar significativamente la complejidad del sistema embebido.

En contraste, para la herramienta de asistencia con inferencia en tiempo real, se selecciona el STM32 presente en el Arduino Portenta H7, aprovechando su doble núcleo (Cortex-M7 y Cortex-M4), unidad de punto flotante y capacidades DSP. Estas características le permiten

procesar datos de múltiples sensores de manera simultánea y ejecutar algoritmos de inferencia neuromórfica con baja latencia, cumpliendo los requerimientos de respuesta en tiempo real necesarios en aplicaciones de asistencia a adultos mayores. Además, su elevado número de pines y soporte para interfaces modernas garantiza la integración de módulos de comunicación y periféricos adicionales.

En conclusión, la combinación del ATmega328P para el registro de datos y del Portenta H7 (STM32) para el procesamiento y la inferencia en tiempo real representa un equilibrio óptimo entre eficiencia energética, capacidad de procesamiento y escalabilidad, asegurando la viabilidad del sistema wearable tanto en la etapa de desarrollo de datos como en su implementación final como herramienta de asistencia. Dadas las capacidades del microcontrolador seleccionado para el Sistema neuro mórfico es necesario resaltar la unidad sensorial que conformará el Sistema, compuesta por diferentes tipos de sensores.

Los sensores son dispositivos esenciales en la medición y conversión de señales físicas en sistemas electrónicos. Un sensor es un dispositivo que detecta una magnitud física, como temperatura, presión o luz, y la convierte en una señal medible.

El desarrollo de sensores inteligentes ha impulsado el crecimiento de tecnologías como el Internet de las Cosas (IoT), la automatización industrial y los sistemas de asistencia en tiempo real. Gracias a su capacidad para auto calibrarse, detectar fallos y comunicarse mediante protocolos avanzados, estos sensores están revolucionando múltiples sectores y abriendo nuevas posibilidades en la recolección y análisis de datos.

Dentro de las aplicaciones de monitoreo y reconocimiento de actividades humanas, históricamente ha predominado el uso de unidades de medición inercial y de interfaces capacitivas para la detección de eventos anómalos. En particular, los sensores capacitivos de la familia Atmel QT se han posicionado como una opción popular en el desarrollo de sistemas interactivos y dispositivos de asistencia. Esta popularidad se debe a su robustez en la detección táctil, facilidad de integración con microcontroladores y versatilidad para implementar teclados, botones o superficies sensibles al tacto en entornos portátiles y de bajo consume, tanto en aplicaciones rígidas como flexibles enfocadas a sistemas wearables.

A continuación en la Tabla 3 se presenta una table comparative de diferentes modelos de sensores capacitivos con diferentes capacidades y principios de funcionamiento.

Tabla 1.3: Comparativa de sensores capacitivos de la familia AT42QT.

Sensor	N° de entradas	Tipo de detección	Interfaz	Consumo aproximado
AT42QT1070	7	Autocapacitancia	I2C	200 μ A (modo activo)
AT42QT2120	12	Autocapacitancia	I2C / SPI	300 μ A (modo activo)
AT42QT2160	16	Autocapacitancia / Capacitancia mutua	I2C / SPI	400 μ A (modo activo)
AT42QT60248	Hasta 248	Autocapacitancia / Capacitancia mutua	SPI / Matrix Scanning	1–5 mA (dependiendo del número de canales activos)

En primer lugar, se observa que el número de entradas varía considerablemente: desde modelos básicos como el AT42QT1070 (7 entradas) o el AT42QT2160 (16 entradas), hasta dispositivos de alta densidad como el AT42QT60248, que admite hasta 248 canales, orientado a sistemas de detección masiva o pantallas táctiles. Esto refleja que la familia QT está diseñada para cubrir tanto aplicaciones compactas como escalables.

Respecto al tipo de detección, la mayoría de los modelos (QT1070, QT2160) trabajan bajo el principio de auto capacitancia, ideal para botones y superficies táctiles independientes. Sin embargo, algunos dispositivos avanzados como el AT42QT60248 permiten implementar capacitancia mutua, lo que habilita el diseño de matrices táctiles y paneles más complejos, incrementando la versatilidad de uso en interfaces modernas.

En cuanto a la interfaz de comunicación, los sensores más simples ofrecen principalmente I2C, lo que los hace fáciles de integrar con microcontroladores de bajo consumo. Modelos más

avanzados incluyen soporte adicional para SPI, ampliando la compatibilidad y velocidad en sistemas embebidos de mayor rendimiento.

El consumo energético es otra característica distintiva: los sensores básicos como el QT1070 están optimizados para bajo consumo (del orden de μA a pocos mA), siendo adecuados para sistemas portátiles o alimentados por batería. Los modelos con mayor número de entradas, como el QT60248, naturalmente presentan un consumo mayor, pero aún mantienen eficiencia energética gracias a modos de ahorro de energía y configuraciones programables.

La configuración programable es una constante en todos los dispositivos, aunque con diferente nivel de sofisticación: mientras que los modelos básicos permiten ajustes de sensibilidad y calibración, los más avanzados incorporan configuraciones detalladas de tiempos de escaneo, detección multicanal y algoritmos de supresión de ruido. Esto otorga mayor control al diseñador sobre el comportamiento del sistema táctil.

En términos de compatibilidad, toda la familia mantiene una buena integración con microcontroladores Atmel/Microchip y plataformas IoT, siendo ampliamente utilizada en aplicaciones de consumo, industriales y médicas.

Para el propósito de la propuesta de solución a desarrollar en este trabajo se seleccionó el AT42QT1070 ya que es un sensor de autocapacitancia de bajo consumo con capacidad para procesar hasta 7 canales de pads capacitivos.

Por otra parte, para el reconocimiento de actividades humanas (HAR) por sus siglas en inglés Human Activity Recognition, existen diversas unidades de medición que brindan un monitoreo de las actividades del usuario, sin embargo, los más populares y efectivos hasta el momento son los acelerómetros con giroscopio. Que gracias a una correcta posición del sensor se puede tener un monitoreo de la posición del usuario en estado de reposo, y al realizar actividades cotidianas como caminar, sentarse, levantarse, agacharse o incluso al sufrir una caída.

A continuación, se muestra una tabla comparativa de algunas de las unidades de medición inercial más populares en el desarrollo de herramientas de reconocimiento de actividad humana.

Tabla 1.4: Comparativa de sensores inerciales utilizados en sistemas embebidos.

Sensor	Ejes	Rango acelerómetro	Rango giroscopio	Consumo típico
MPU6050	6 (3 ace + 3 giro)	$\pm 2g$ / $\pm 4g$ / $\pm 8g$ / $\pm 16g$	± 250 / ± 500 / ± 1000 / ± 2000 °/s	3.9 mA
MPU9250	9 (3 ace + 3 giro + 3 mag)	$\pm 2g$ / $\pm 4g$ / $\pm 8g$ / $\pm 16g$	± 250 / ± 500 / ± 1000 / ± 2000 °/s	3.5 mA
LSM6DS3	6 (3 ace + 3 giro)	$\pm 2g$ / $\pm 4g$ / $\pm 8g$ / $\pm 16g$	± 125 / ± 245 / ± 500 / ± 1000 / ± 2000 °/s	0.9 mA (ultra low power)
ICM-20948	9 (3 ace + 3 giro + 3 mag)	$\pm 2g$ / $\pm 4g$ / $\pm 8g$ / $\pm 16g$	± 250 / ± 500 / ± 1000 / ± 2000 °/s	2.5 mA
ADXL345	3 ace	$\pm 2g$ / $\pm 4g$ / $\pm 8g$ / $\pm 16g$	—	140 μ A

El análisis comparativo de sensores inerciales permite identificar aquellos que ofrecen un balance adecuado entre precisión, consumo energético, facilidad de integración y costo. En este contexto, se observa que los sensores de nueve ejes, como el MPU9250 o el ICM-20948, proporcionan información más completa al integrar un magnetómetro, lo cual es útil para aplicaciones de orientación absoluta. Sin embargo, esta característica implica un mayor costo y complejidad de procesamiento, lo cual no resulta estrictamente necesario para el reconocimiento de actividades humanas en ambientes controlados.

En contraste, el MPU6050 destaca como una solución ampliamente adoptada en la comunidad científica y en el desarrollo de sistemas embebidos. Su combinación de acelerómetro y giroscopio de seis ejes, junto con su interfaz de comunicación sencilla mediante I²C y su compatibilidad con múltiples plataformas (Arduino, ESP32, Raspberry, STM32), lo convierten en un dispositivo ideal tanto para la fase experimental como para la implementación final del sistema neuromórfico; es un sensor de bajo costo que integra ambas funcionalidades en un mismo encapsulado con un consumo moderado, ideal para aplicaciones portátiles.

A continuación, se presentan las bases conceptuales para el desarrollo de un sistema neuromórfico mediante las unidades mencionadas anteriormente.

1.7 ASPECTOS GENERALES PARA EL DESARROLLO DE UN SISTEMA NEUROMÓRFICO

1.7.1 INTELIGENCIA ARTIFICIAL

La inteligencia artificial es una rama de la tecnología que permite a las computadoras y máquinas simular el aprendizaje humano, la comprensión, la resolución de problemas, la toma de decisiones, la creatividad e incluso la autonomía.[26]

Las aplicaciones y dispositivos equipados con IA pueden ver e identificar información nueva.

En 2024, la mayoría de los investigadores y profesionales de la Inteligencia Artificial se centran en avances en la IA generativa. A continuación, en la siguiente figura se muestra una relación breve de la inteligencia artificial con el Machine Learning, el Deep Learning y la IA Generativa:

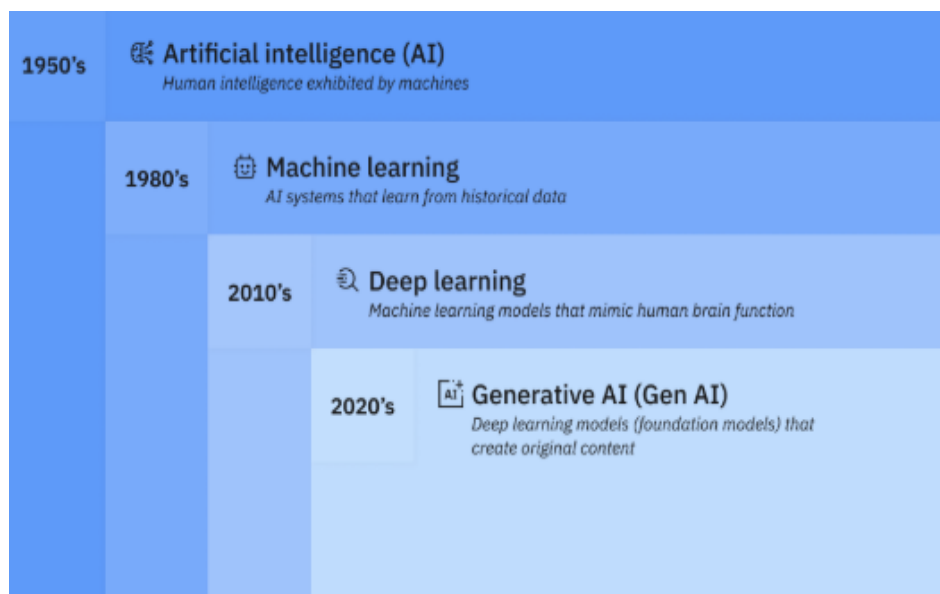


Figura 1.1: Esquema representativo de la evolución de la Inteligencia Artificial.

Si bien el concepto de inteligencia artificial existe desde los años 50, el término Inteligencia Artificial ganó auge a partir del año 2020, cuando especialistas iniciaron la revolución de la IA

generativa.

1.7.2 MACHINE LEARNING Y DEEP LEARNING

Si observamos la [Figura 1.1](#) podemos analizar que la Rama del Machine Learning deriva del concepto de Inteligencia Artificial a partir de los años 80.

El Machine Learning consiste en crear modelos entrenando a un algoritmo para hacer predicciones o tomar decisiones basadas en datos. Abarca una amplia gama de técnicas que permiten a las computadoras aprender y hacer inferencias basadas en datos sin estar programadas explícitamente para tareas específicas. [26]

Dentro del Machine Learning, existen diversos tipos de técnicas o algoritmos, como la regresión lineal, árboles de decisión, bosques aleatorios, máquinas de vectores de soporte (SVM), clustering, entre otros.

La selección del algoritmo a utilizar depende del enfoque o el problema que se quiera solucionar y de los datos que se dispongan. Dentro de la rama del Machine Learning existen dos tipos de aprendizaje, el aprendizaje supervisado y el aprendizaje no supervisado.

Aprendizaje supervisado:

- En el aprendizaje supervisado, se proporciona al algoritmo un conjunto de datos etiquetados, donde cada entrada tiene una salida deseada.
- El algoritmo aprende la relación entre las entradas y las salidas para poder predecir la salida para nuevas entradas no vistas.
- Ejemplos de tareas de aprendizaje supervisado incluyen clasificación (categorizar datos) y regresión (predecir valores continuos).
- Algunos algoritmos comunes de aprendizaje supervisado son árboles de decisión, máquinas de vectores de soporte y redes neuronales.

Aprendizaje supervisado:

- En el aprendizaje no supervisado, el algoritmo trabaja con datos sin etiquetar y debe descubrir patrones y relaciones por sí mismo.
- El objetivo es encontrar estructuras ocultas en los datos, como agrupamientos (clustering) o dimensiones reducidas.
- Ejemplos de tareas de aprendizaje no supervisado incluyen clustering (agrupamiento de datos similares) y reducción de dimensionalidad.
- Algunos algoritmos comunes de aprendizaje no supervisado son k-means, PCA (Principal Component Analysis) y redes neuronales autoencoders.

El Deep Learning (aprendizaje profundo) es una subárea del Machine Learning que utiliza redes neuronales artificiales con múltiples capas (profundas) para modelar y aprender representaciones jerárquicas de datos. Estas redes son capaces de identificar patrones complejos en grandes volúmenes de información, extrayendo características automáticamente a medida que la información fluye desde las capas de entrada hasta las de salida. Gracias a esta capacidad, el Deep Learning ha permitido avances significativos en tareas como el reconocimiento de voz, la visión por computadora, el procesamiento del lenguaje natural y la conducción autónoma.^[27]

A diferencia de los algoritmos tradicionales de Machine Learning, que suelen requerir una extracción manual de características, el Deep Learning aprende directamente de los datos crudos (como imágenes, texto o señales) mediante una optimización basada en retropropagación y grandes cantidades de datos etiquetados. Su desempeño ha mejorado notablemente con el uso de GPUs, grandes conjuntos de datos y arquitecturas más sofisticadas, como las redes convolucionales (CNNs), recurrentes (RNNs) o transformadores.

Estas múltiples capas permiten el aprendizaje no supervisado y aportan a la automatización y extracción de características de conjuntos de datos no etiquetados y no estructurados, así como a la realización de sus propias predicciones sobre lo que representan los datos.

Al no requerir intervención humana, el deep learning permite el machine learning a gran escala. Es muy adecuado para el procesamiento de lenguaje natural (PLN), la visión artificial y otras tareas que requieren identificar rápidamente y con precisión patrones y relaciones complejas

en grandes volúmenes de datos.

El deep learning también permite:

- El aprendizaje semisupervisado combina el aprendizaje supervisado y no supervisado mediante el uso de datos etiquetados y no etiquetados para capacitar modelos de IA en tareas de clasificación y regresión.
- El aprendizaje autosupervisado genera etiquetas implícitas a partir de datos no estructurados, en lugar de depender de conjuntos de datos etiquetados para obtener señales de supervisión.
- El aprendizaje por refuerzo se basa en funciones de prueba y error, así como en recompensas, en lugar de extraer información de patrones ocultos.
- Aprendizaje por transferencia, en el que el conocimiento adquirido a través de una tarea o conjunto de datos se emplea para mejorar el rendimiento del modelo en otra tarea relacionada o en un conjunto de datos diferente.

Los enfoques basados en Deep Learning han evidenciado un alto rendimiento en el reconocimiento de actividades humanas; no obstante, sus exigencias en términos de procesamiento y memoria suelen obligar a que el entrenamiento y la inferencia se realicen en plataformas externas o en la nube. Esta dependencia incrementa el tráfico de red de los dispositivos móviles o portátiles y puede comprometer la confidencialidad de los datos personales del usuario.[28]

1.7.3 REDES NEURONALES

Las denominadas Redes Neuronales Artificiales se modelan a partir de la estructura y función del cerebro humano, consta de capas interconectadas de nodos que trabajan juntas para procesar y analizar una serie de datos complejos. Es adecuado para tareas que implican identificar patrones y relaciones complejas en grandes cantidades de datos.

Una red neuronal es un programa o modelo de machine learning que toma decisiones de manera similar al cerebro humano, mediante el uso de procesos que imitan la forma en que las

neuronas biológicas trabajan juntas para identificar fenómenos, sopesar opciones y llegar a conclusiones.[29]

A continuación, en la [Figura 1.2](#) se muestra la estructura de una red neuronal profunda multicapa.

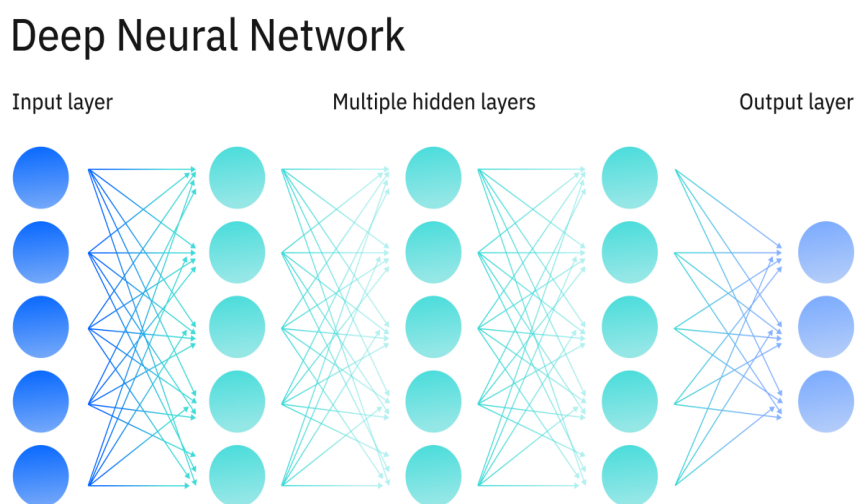


Figura 1.2: Esquema representativo de la evolución de la Inteligencia Artificial.

Cada red neuronal consta de capas de nodos o neuronas. Normalmente, constan de una capa de entrada, una o más capas ocultas y una capa de salida.

Cada nodo se conecta a otros y tiene su propia ponderación y umbral asociados. Si la salida de cualquier nodo está por encima del valor del umbral especificado, ese nodo se activa y envía datos a la siguiente capa de la red. De lo contrario, no se pasa ningún dato a la siguiente capa de la red. Las redes neuronales se basan en datos de entrenamiento para aprender y mejorar su precisión con el tiempo [29]

Con el entrenamiento suficiente, nos permiten clasificar y agrupar datos a alta velocidad, de modo que tareas como el reconocimiento de voz o de imágenes pueden tardar minutos en lugar de horas en comparación con la identificación manual por parte de expertos humanos.

En la literatura, las redes neuronales se denominan redes neuronales artificiales (ANN) o redes

neuronales simuladas (SNN). Son un subconjunto del machine learning y están en el corazón de los modelos de deep learning.

Las redes neuronales se pueden clasificar en diferentes tipos que se utilizan para diferentes propósitos; entre las más comunes se encuentran:

1.7.3.1 REDES NEURONALES DE RETROALIMENTACIÓN

Este tipo de redes también suele denominarse perceptron multicapa (MLP); se compone de una capa de entrada, una o más capas ocultas y una capa de salida.

Es importante señalar que este tipo de red está compuesto por neuronas sigmoideas, no por perceptrones, ya que la mayoría de los problemas del mundo real no son lineales. Por lo general, los datos se ingresan en estos modelos para entrenarlos y constituyen la base de la visión artificial y el procesamiento del lenguaje natural.

1.7.3.2 REDES NEURONALES CONVOLUCIONALES

Son similares a las redes MLP, pero generalmente se utilizan para el reconocimiento de imágenes, el reconocimiento de patrones o la visión artificial. Estas redes aprovechan en particular el álgebra, enfocándose específicamente en la multiplicación de matrices para identificar patrones dentro de una imagen.

1.7.3.3 REDES NEURONALES RECURRENTES

Este tipo de redes neuronales se identifica por sus ciclos de retroalimentación. Estos algoritmos de aprendizaje se aprovechan principalmente cuando se utilizan datos de series de tiempo para hacer predicciones sobre resultados futuros, como predicciones del mercado de valores o pronósticos de ventas.

1.7.4 REDES NEURONALES MULTICAPA

Las Redes Neuronales Multicapa (MLP, por sus siglas en inglés, Multilayer Perceptron) son un tipo de red neuronal feedforward, es decir, una arquitectura en la que la información fluye en

una sola dirección, desde la capa de entrada hacia la capa de salida sin retroalimentación entre las neuronas. [30] Una MLP está compuesta por al menos tres capas de neuronas artificiales: una capa de entrada, una o más capas ocultas y una capa de salida. Cada neurona aplica una función de activación no lineal, lo que permite a la red modelar relaciones complejas y no lineales entre los datos de entrada y las salidas esperadas.

Durante el proceso de entrenamiento, la red ajusta los pesos sinápticos mediante algoritmos de optimización, comúnmente la retropropagación del error (backpropagation), minimizando la diferencia entre las salidas predichas y las reales. Gracias a esta capacidad de aprendizaje supervisado, las MLP son ampliamente utilizadas en tareas de clasificación, regresión y reconocimiento de patrones, siendo una de las arquitecturas más representativas del aprendizaje profundo clásico.

En el contexto de este trabajo, se utiliza una red neuronal MLP como núcleo del sistema neuromórfico de reconocimiento de actividades humanas, debido a su eficacia para identificar patrones temporales en los datos inerciales obtenidos de sensores como el acelerómetro y el giroscopio. Su estructura jerárquica y su capacidad para generalizar a partir de los datos de entrenamiento permiten realizar inferencias precisas en tiempo real, optimizando el balance entre complejidad computacional y rendimiento energético, aspectos esenciales en sistemas embebidos portátiles.

En este capítulo se establecieron los fundamentos teóricos, conceptuales y tecnológicos que sustentan el desarrollo del sistema neuromórfico propuesto, abordando la problemática asociada a la detección de caídas en adultos mayores, así como la justificación de su relevancia social y científica. Se definieron los objetivos e hipótesis que orientan el trabajo, además de revisarse el estado del arte en tecnologías de monitoreo, procesamiento de señales inerciales y enfoques basados en inteligencia artificial aplicados a entornos geriátricos. Con base en este marco de referencia, se consolidaron los criterios de diseño y las directrices técnicas que guiarán la implementación del sistema. En el siguiente capítulo se detalla la metodología empleada para el desarrollo del prototipo, incluyendo las etapas de diseño electrónico, integración del hardware, procesamiento de datos y validación experimental del modelo neuromórfico.

CAPÍTULO 2: DESCRIPCIÓN DE METODOLOGÍA PARA EL DESARROLLO DE UNA HERRAMIENTA DE ASISTENCIA BASADA EN UN SISTEMA NEUROMÓRFICO ORIENTADO AL RECONOCIMIENTO DE ACTIVIDADES EN ADULTOS MAYORES

En este capítulo se describe la metodología para el diseño, implementación y validación de un sistema embebido de monitoreo de actividades humanas, enfocado a una herramienta de asistencia para la detección de caídas destinada a usuarios de la tercera edad. Esta tecnología surge como una herramienta para cubrir de manera parcial el desgaste del cuidador primario y mejorar la calidad de vida de las personas de la tercera edad, monitoreando uno de los síndromes geriátricos más comunes en este grupo: las caídas.

El presente proyecto es de tipo científico, con una metodología experimental para evaluar el comportamiento de los sensores capacitivos matriciales, buscando el desarrollo de herramientas de asistencia para adultos mayores, como se describe en los objetivos del presente trabajo.

Para obtener un producto final eficaz, se llevará a cabo una metodología por etapas definidas que implicará habilidades y técnicas a desarrollar para obtener los diferentes productos de cada uno de los objetivos específicos, con el fin de lograr una integración confiable del dispositivo.

En la [Figura 2.1](#) se muestra un diagrama a bloques de la metodología general para el desarrollo del sistema neuromórfico orientado al reconocimiento de actividades en adultos mayores.

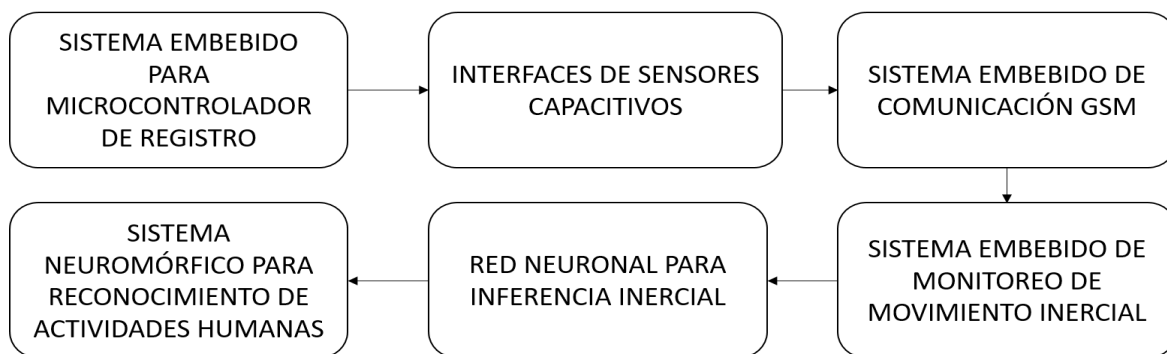


Figura 2.1: Metodología para el desarrollo del proyecto de investigación etapa 1.

Como se puede analizar en el diagrama anterior, la metodología para el desarrollo del presente

trabajo de investigación se divide en seis etapas. La primera etapa consiste en el desarrollo del sistema embebido para el registro y procesamiento de datos. La etapa 2 consiste en el desarrollo de interfaces capacitivas para el procesamiento de señales mediante un sistema embebido. Posteriormente, en la etapa 3 se desarrolla, caracteriza y configura un sistema embebido para la gestión de señales de comunicación GSM, tanto para el envío y recepción de mensajes SMS, como para el envío y recepción de llamadas telefónicas. En la etapa 4 se desarrolla una de las metodologías cruciales del presente trabajo, que es el desarrollo de un sistema embebido para la unidad de medición inercial para el registro de la base de datos. Posteriormente, en la etapa 5, se crea y entrena el modelo de Red Neuronal para la inferencia en datos inerciales categóricos. Finalmente, en la etapa 6 se realiza la integración de cada uno de los resultados de las demás etapas en un sistema embebido para adaptar el sistema neuromórfico al reconocimiento de actividades humanas en adultos mayores.

Cada una de estas etapas se describe a detalle a continuación.

A continuación se muestra en la [Figura 2.2](#) la etapa 1 de la metodología para el desarrollo del trabajo de investigación.

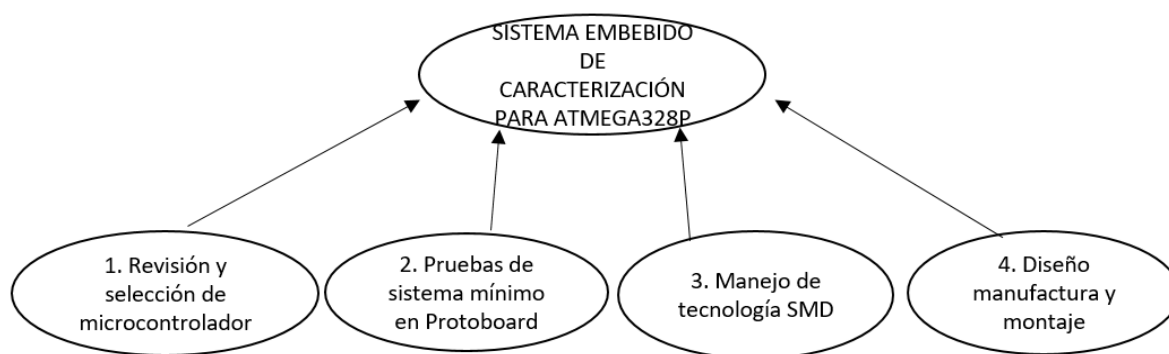


Figura 2.2: Metodología para el desarrollo del proyecto de investigación etapa 1.

Como primera instancia de la metodología del proyecto, se propone el microcontrolador a utilizar para programar los diferentes sistemas embebidos que se desarrollarán para la caracterización de los sensores y módulos de comunicación.

A continuación, se revisa la hoja de datos de cada uno de los componentes necesarios para el sistema mínimo del microcontrolador seleccionado.

Se elabora un diagrama esquemático del sistema mínimo del microcontrolador con el propósito de establecer las conexiones de cada uno de los componentes del sistema.

Posteriormente, se realizan una serie de pruebas preliminares para comprobar el funcionamiento del sistema mínimo y del microcontrolador.

Una vez validado el funcionamiento, se procede al diseño y la manufactura del sistema embebido con las conexiones establecidas en el paso anterior.

Se valida el funcionamiento del sistema embebido y se realizan los pasos de configuración del microcontrolador necesarios para proceder a la carga del código.

Posteriormente, se procede a la etapa 2 de la metodología, la cual consiste en el desarrollo de interfaces capacitivas con el objetivo de elaborar una interfaz de detección enfocada a la activación de funciones de comunicación GSM, como herramienta de asistencia para adultos mayores.

A continuación, en la [Figura 2.3](#) se muestra el diagrama de la etapa 2 del desarrollo del presente trabajo de investigación.

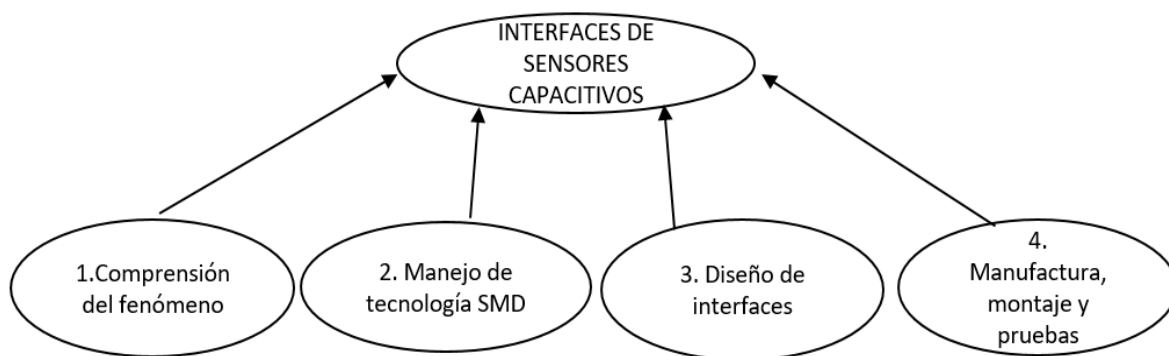


Figura 2.3: Metodología para el desarrollo del proyecto de investigación etapa 2.

Como primera instancia, se debe comprender el fenómeno de capacitancia; esto se llevará a cabo mediante la revisión de referencias bibliográficas específicas.

Mediante la revisión de las referencias bibliográficas, se analizarán las diferentes geometrías para el diseño de los sensores capacitivos matriciales, tomando en cuenta criterios como área de detección, campo electromagnético, viabilidad en cuanto a espacio, densidad de la

geometría, entre otros.

Uno de los puntos clave del diseño y manufactura de tarjetas de circuito impreso es la manipulación de tecnología de montaje superficial (SMD) con el objetivo de optimizar espacios en las tarjetas de circuito impreso para sistemas embebidos que están enfocados a herramientas portátiles.

El objetivo principal de la primera etapa del presente trabajo consiste en desarrollar interfaces capacitivas vinculadas a su vez, a una interfaz visual programada en una computadora personal para comprender y observar el funcionamiento de los sensores capacitivos de auto-capacitancia. Dicha interfaz se presenta en la discusión de resultados en el capítulo 4.

Es crucial conocer las diferentes herramientas y técnicas en los procesos de manufactura de tarjetas de circuito impreso para poder desarrollar un sistema con un grado de confiabilidad sobresaliente; por ello, fue necesario perfeccionar la técnica de grabado y establecer los valores de las variables involucradas en los procesos de manufactura, como el tiempo de reacción del cloruro férrico, la temperatura en el proceso de transferencia térmica, entre otros.

Para continuar con la segunda etapa de la metodología para el desarrollo del proyecto de investigación, en la [Figura 2.4](#) se muestra el diagrama para el Sistema Embebido de Comunicación GSM:

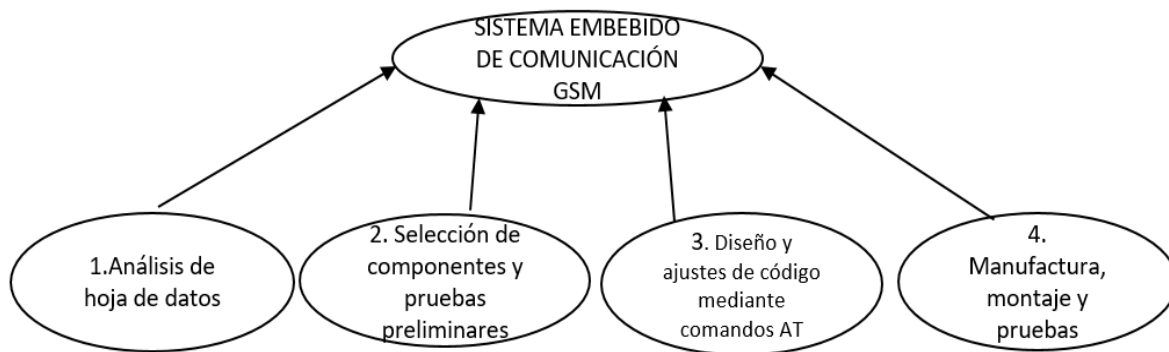


Figura 2.4: Metodología para el desarrollo del proyecto de investigación etapa 3.

Como primera instancia y al igual que en la etapa pasada, el primer paso para elaborar un sistema es conocer la teoría y las hojas de datos de los componentes que se utilizarán para el desarrollo.

Una vez justificada la selección de los componentes, se procede a realizar pruebas preliminares en Protoboard, para validar el funcionamiento de cada uno de los componentes seleccionados y evaluar si es el desempeño que se busca para la aplicación.

Una vez validado el funcionamiento de los componentes seleccionados, se procede a realizar ajustes en caso de ser necesario y se realiza el diseño del código para la configuración del módulo de comunicación en sus modos de mensajería, de llamada y en su modo de reposo, con el objetivo de cuidar el desempeño de la batería en el prototipo final.

Finalmente se realiza el diseño del sistema embebido para hacer pruebas finales del módulo de comunicación GSM para evaluar la calidad de llamada, la transmisión de mensajería SMS y el consumo de corriente del módulo, tanto en transmisión como en modo de reposo.

A continuación, en la siguiente [Figura 2.5](#) se muestra el diagrama de la metodología a desarrollar para el diseño y manufactura del sistema embebido para registro de datos inerciales.

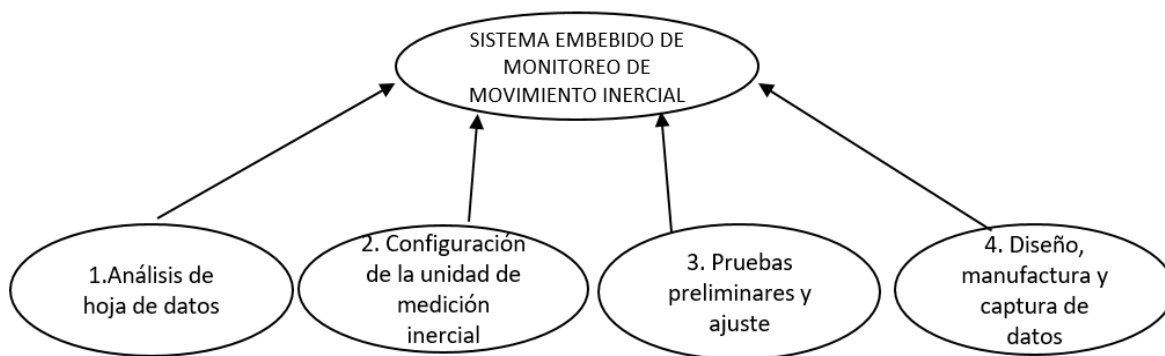


Figura 2.5: Metodología para el desarrollo del proyecto de investigación etapa 4.

Continuando con una de las etapas centrales del desarrollo del proyecto de investigación, se presenta la metodología para el desarrollo del sistema embebido para registro de datos inerciales, con el objetivo de tener en posesión una base de datos inerciales bien estructurada.

Para iniciar con esta etapa se comenzó por revisar las hojas de datos del acelerómetro con giroscopio a utilizar, así como su programación, su sensibilidad, su orientación y, principalmente, sus modos de configuración.

Además, se hizo una revisión exhaustiva en diversas fuentes bibliográficas para seleccionar los

movimientos cotidianos más representativos en la vida de un adulto mayor, siendo en este caso 5 movimientos representativos, de los cuales 3 se clasificaron en 3 subcategorías dependiendo de la velocidad con la que se ejecutaba el movimiento.

Posteriormente se realizaron pruebas preliminares para identificar la mejor configuración tanto para el acelerómetro como para el giroscopio mediante pruebas preliminares de cada uno de los movimientos seleccionados.

Finalmente se diseña y manufactura el sistema embebido para el registro de datos inerciales y se presenta la herramienta para el registro y preprocesamiento de los datos.

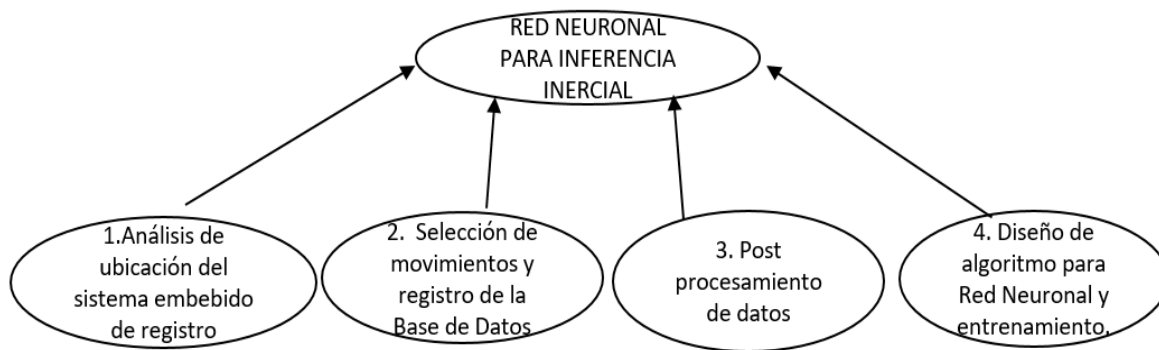


Figura 2.6: Metodología para el desarrollo del proyecto de investigación etapa 5.

Para continuar con la metodología para el desarrollo de la Red Neuronal para realizar inferencia en datos inerciales, fue necesario evaluar la colocación del sistema en la ubicación más óptima, realizando de manera exhaustiva una revisión en las diferentes fuentes bibliográficas del estado del arte, así como en fuentes clínicas. Esto debido a que la ubicación del sistema en prototipos wearables juega un papel muy importante, no solo para el desempeño y calidad de los datos registrados, sino para asegurar la salud del usuario al utilizar la herramienta wearable.

Por otra parte, en esta etapa se consideran los entornos y condiciones para el registro de la base de datos para el proyecto de investigación y se procede con la metodología de captura de datos.

Dado que en el microcontrolador se programaron interrupciones para activar el registro de datos, los pasos a seguir para iniciar el registro serán los siguientes: enlazar el sistema con la

herramienta PLX-DAQ de Excel y conectar los puertos seriales mediante el módulo Bluetooth del sistema embebido; posteriormente, apretar el botón de reset del microcontrolador para iniciar una nueva medición, y en cuanto el sistema detecte que el umbral de aceleración se superó, entonces transmitirá los datos inerciales durante un periodo de tiempo programado anteriormente. Finalmente, se detiene la transmisión de datos y se vuelve a reiniciar el microcontrolador para iniciar una nueva medición.

Una vez con la base de datos completa con los datos inerciales registrados, se procederá a cargar la base de datos a algoritmos de formato necesarios para ingresar al entrenamiento de la red neuronal, donde los datos son acomodados de manera estratégica para facilitar la lectura de datos por el algoritmo de Deep Learning, así como su etiquetado por clases.

En caso de ser necesario, se utilizarán técnicas de reducción de datos para hacer el modelo de Deep Learning más compacto y enfocar hacia un microcontrolador; en este caso, el Arduino Portenta H7 Lite.

Finalmente, mediante un entorno virtual dedicado a la tarjeta gráfica de una computadora personal mediante Anaconda3, se descargan las librerías necesarias para la creación de la Red Neuronal, como TensorFlow, Keras, Pandas, Seaborn, y algunas complementarias como Matplotlib para realizar los gráficos de resultados.

Para esta etapa se pretende evaluar distintos modelos de Redes Neuronales con diferente número de entradas, tomando en cuenta variables como la precisión y la pérdida en entrenamiento y en prueba de cada uno de ellos, con el objetivo de seleccionar aquel que cuente con un valor de precisión válido y sea lo suficientemente compacto para agregarse al microcontrolador del prototipo.

Por último, se presenta en la [Figura 2.7](#) la metodología para integrar el prototipo final de la herramienta de asistencia para alerta de caídas en adultos mayores.

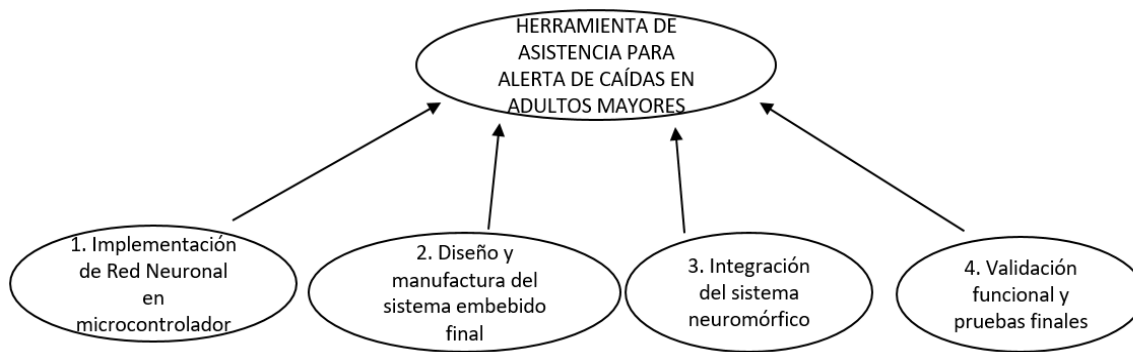


Figura 2.7: Metodología para el desarrollo del proyecto de investigación etapa 6.

Para la última etapa de la metodología se selecciona el modelo de Red Neuronal con el mejor desempeño obtenido tomando en cuenta datos como la precisión en entrenamiento y en prueba.

Posteriormente se elabora un algoritmo para extraer los parámetros y características del modelo de Red Neuronal en un formato compatible para ser ingresado al código del microcontrolador.

Se realiza el diagrama esquemático del prototipo final tomando en cuenta cada uno de los sistemas desarrollados en las etapas anteriores, integrando cada módulo, cada sensor y cada conexión en un mismo sistema. La elaboración

Para ello, se siguen los lineamientos establecidos por normas internacionales como la IEC 61082-1 (Preparation of documents used in electrotechnology – Part 1: Rules) y la IEEE Std 315-1975 (Graphic Symbols for Electrical and Electronics Diagrams), que establecen criterios para la representación gráfica, simbología, interconexión y disposición de los elementos en un diagrama.

Finalmente se procede con el diseño de la tarjeta de circuito impreso (PCB) para el prototipo final de esta tesis, que responde a la necesidad de consolidar y optimizar la integración de los componentes electrónicos del sistema, garantizando su funcionalidad, confiabilidad y facilidad de montaje.

La implementación en PCB permite reducir las dimensiones del sistema, minimizar el ruido eléctrico y mejorar la robustez mecánica en comparación con conexiones mediante protoboards o cables sueltos.

Asimismo, el diseño del PCB se fundamenta en buenas prácticas de ingeniería electrónica y normas internacionales como la IPC-2221 (Generic Standard on Printed Board Design), que establecen criterios para el diseño eléctrico, la disposición de componentes, el ruteo de pistas, la gestión térmica y la protección contra interferencias electromagnéticas. Estas normas aseguran que la tarjeta cumpla con los requerimientos técnicos y ambientales, así como con las especificaciones funcionales del prototipo.

Finalmente se valida el funcionamiento de la herramienta de asistencia y se elabora una serie de pruebas finales del prototipo.

En este capítulo se describe la metodología propuesta para el diseño e implementación del sistema neuromórfico, abarcando las etapas de selección de componentes, integración electrónica, adquisición y procesamiento de datos, así como el entrenamiento y validación del modelo de inferencia. La metodología propuesta permitió establecer una ruta estructurada que garantiza la coherencia entre los objetivos planteados y el desarrollo técnico del sistema, asegurando la reproducibilidad del proceso y la consistencia de los resultados. En el siguiente capítulo se presenta el desarrollo experimental, donde se detalla la aplicación práctica de cada una de las etapas metodológicas, así como la obtención y análisis de los datos experimentales que sustentan la validación del sistema.

CAPÍTULO 3: DESARROLLO EXPERIMENTAL PARA IMPLEMENTACIÓN DE UN SISTEMA NEUROMÓRFICO DE RECONOCIMIENTO DE ACTIVIDADES EN HERRAMIENTA DE ASISTENCIA PARA ADULTOS MAYORES

En este capítulo se presenta la metodología empleada para la integración del sistema neuromórfico destinado al reconocimiento de actividades en adultos mayores. Se detallan los procedimientos para la adquisición y organización de datos inerciales, la selección y el procesamiento de características relevantes, así como el diseño del modelo de inferencia utilizado. Esta estructura metodológica establece el fundamento técnico necesario para la construcción de cada una de las unidades que conforman el sistema y prepara el marco con el que posteriormente se evaluará su desempeño durante la fase de validación.

El presente proyecto es de tipo científico, con una metodología experimental para evaluar el comportamiento de los sensores inerciales en el reconocimiento de actividades humanas mediante un sistema neuromórfico entrenado con un modelo de Redes Neuronales de tipo secuencial para categorización, buscando el desarrollo de herramientas de asistencia para adultos mayores, como la que se describe en los objetivos del presente trabajo.

Para obtener un producto final eficaz, se llevará a cabo una metodología por etapas definidas que implicarán habilidades y técnicas a desarrollar para obtener los diferentes resultados propuestos de cada una de las etapas, llegando al final a una integración de cada uno de ellos para conformar el producto final.

3.1 METODOLOGÍA PARA LA PROGRAMACIÓN Y MONTAJE DE UN MICROCONTROLADOR ATMEGA328P-AU (Etapa 1)

La primera etapa de la metodología se centra en el análisis de los componentes clave y la selección del microcontrolador, así como en su configuración inicial, con el propósito de establecer una base sólida para el desarrollo del sistema embebido. Este proceso es fundamental, ya que las decisiones tomadas en esta fase impactarán directamente en el rendimiento, la escalabilidad y la viabilidad técnica del proyecto.

En esta etapa se contemplan cuatro subetapas que permiten abordar de manera estructurada la elección y validación del microcontrolador. En primer lugar, se realiza la revisión de la hoja de datos del microcontrolador, con el objetivo de comprender a profundidad sus características técnicas, capacidades de procesamiento, periféricos disponibles y requisitos eléctricos, lo cual facilita su correcta integración en el sistema. Posteriormente, se lleva a cabo la prueba de un sistema mínimo en protoboard, que permite validar el funcionamiento básico del microcontrolador y asegurar la correcta alimentación, programación y comunicación antes de avanzar a un diseño más complejo.

Dado que el proyecto requiere componentes de tecnología de montaje superficial (SMD), se incluye una subetapa dedicada al desarrollo de habilidades y técnicas para su manejo, considerando que este tipo de componentes permite optimizar el tamaño y la eficiencia del diseño final. Finalmente, se realiza el diseño, manufactura y montaje del sistema embebido, integrando todos los elementos validados en las fases previas para obtener una tarjeta funcional y confiable, adecuada para las pruebas y desarrollos posteriores.

Esta primera etapa garantiza que el sistema se construya sobre una base técnica robusta, reduciendo riesgos en etapas avanzadas y asegurando la correcta selección y configuración del microcontrolador como el sistema encargado del procesamiento de señales.

3.1.1 Revisión y selección de microcontrolador

Como se analizó en el capítulo 1, la [Tabla 1.2](#) muestra la información detallada de los microcontroladores más destacados para los propósitos del presente proyecto, destacando el ATmega328P como el microcontrolador de bajo consumo y bajo costo, ideal para aplicaciones portátiles sencillas.

Por otra parte, el STM32H747 que contiene la tarjeta de desarrollo Arduino Portenta H7 es una de las mejores opciones para realizar aplicaciones de IA embebida, para el procesamiento intensivo de datos y tareas en tiempo real debido a su doble núcleo y su amplia memoria Flash para soportar modelos pesados.

Considerando el análisis previo y las características de cada microcontrolador, se determinó

una estrategia dual de implementación para optimizar el desarrollo del proyecto. En la fase inicial, se seleccionó el ATmega328P como núcleo para la construcción de los sistemas embebidos de prueba, enfocándose en la configuración y validación de los sensores, así como en el registro y almacenamiento de las bases de datos de entrenamiento, gracias a su bajo consumo energético y facilidad de integración con hardware básico.

Por otro lado, para la etapa final, se optó por utilizar la tarjeta Arduino Portenta H7, basada en el STM32H747, la cual ofrece la capacidad de procesamiento necesaria para implementar un sistema neuromórfico en tiempo real, ejecutando modelos de inteligencia artificial embebida con mayor eficiencia y precisión. Esta combinación de microcontroladores permite una transición gradual desde la experimentación y validación inicial hacia una solución final robusta y de alto desempeño.

3.1.2 Prueba de sistema mínimo en Protoboard

Para validar el desarrollo de la herramienta de asistencia se desarrolló el diagrama esquemático de un sistema embebido del circuito funcional para programar el microcontrolador ATmega328P seleccionado para este proyecto de investigación.

A continuación, en la [Figura 3.1](#) se muestra el diagrama esquemático de conexiones del sistema mínimo para el microcontrolador ATmega328P-AU.

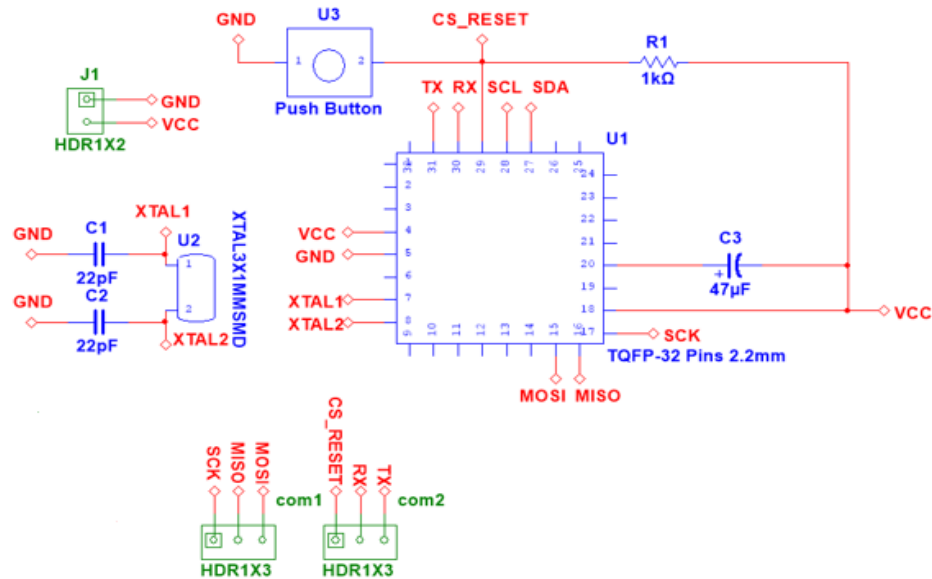


Figura 3.1: Diagrama esquemático de conexiones para programación del microcontrolador ATmega328P-AU

Como se puede observar en la imagen, se utilizarán pines para poder realizar el quemado del Bootloader al microcontrolador, así como pines para poder cargar los respectivos códigos de prueba.

A continuación se realizaron las respectivas conexiones del sistema mínimo en Protoboard para realizar el quemado del bootloader mediante los pines de comunicación serial SPI y la señal de reloj para sincronización (SCK).

Para realizar las conexiones al microcontrolador ATmega328P-AU se utilizó un socket de 32 pines para poder conectar al Protoboard y realizar el quemado del Bootloader y las pruebas específicas.

Una de las razones por las que se realizaron las pruebas de conexiones del sistema mínimo en Protoboard fue para validar, como primera instancia, el funcionamiento del dispositivo, así como comprobar el funcionamiento independiente de cada componente que conformaría el sistema.

Para poder cargar los códigos al microcontrolador se utilizó un convertidor USB-TTL, específicamente el modelo CP2102. Sin embargo, como paso preliminar, fue necesario cargar el

Bootloader del microcontrolador utilizando una tarjeta de desarrollo Arduino UNO, como se puede ver a continuación en la [Figura 3.2](#).

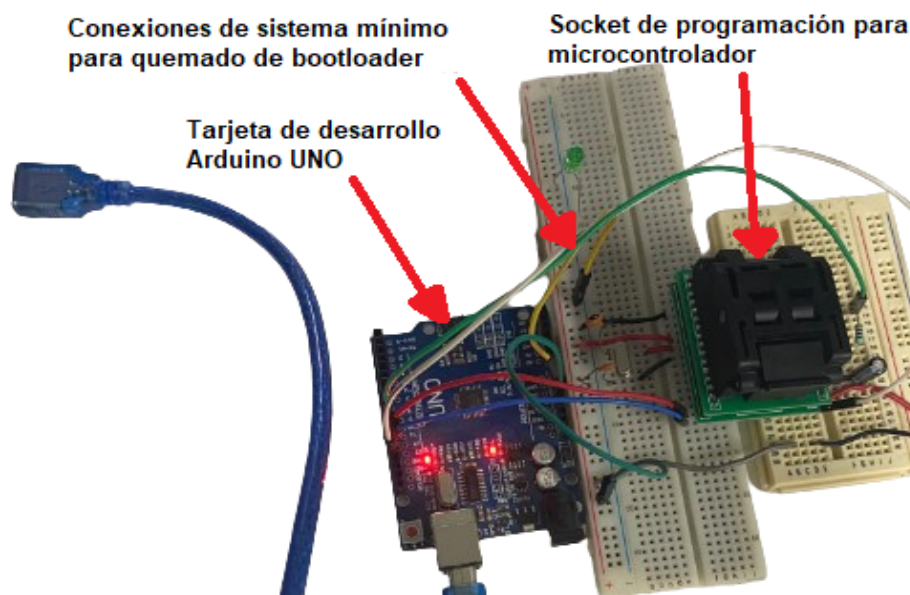


Figura 3.2: Conexión de sistema mínimo para carga de Bootloader de microcontrolador ATmega328P-AU.

Una vez cargado el Bootloader al microcontrolador ATmega328P-AU, se optó por conseguir un convertidor de comunicación USB-Serial como el CP2102, que se utilizó como primera prueba de carga de código para microcontroladores.

Realizando las respectivas conexiones con el convertidor y el microcontrolador, se realizó una prueba de carga de un código simple para encender y apagar un diodo LED, como se puede ver a continuación en la [Figura 3.3](#).

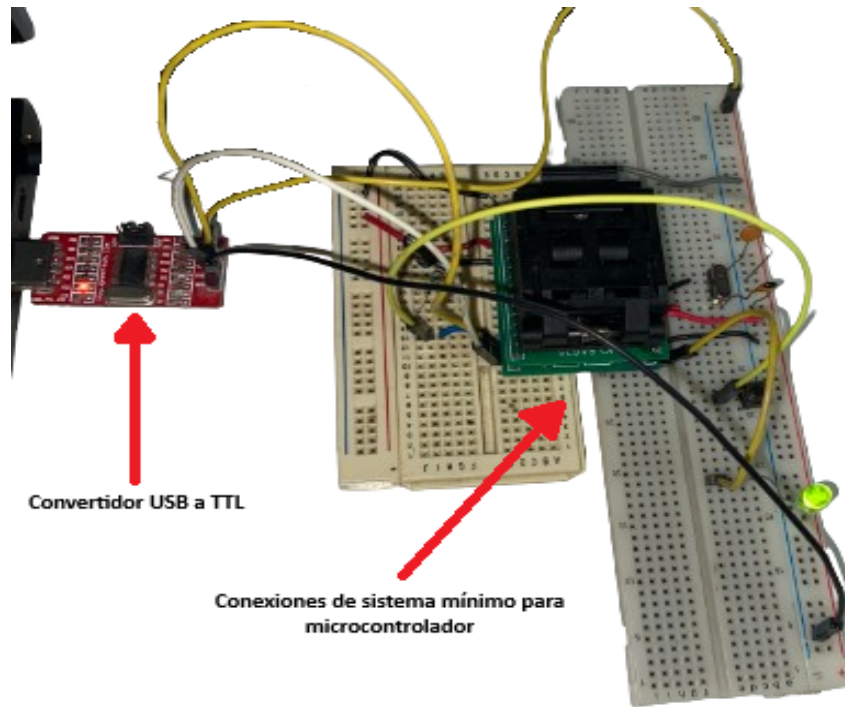


Figura 3.3: Conexión de sistema mínimo para carga de código en microcontrolador ATMega328P-AU.

Una vez validado el sistema mínimo para el funcionamiento del microcontrolador se optó por desarrollar una metodología para la selección de componentes del sistema embebido para el microcontrolador.

3.1.3 Manejo de tecnología SMD

Con el objetivo de desarrollar sistemas portátiles se seleccionaron los componentes de montaje superficial, debido a sus características por ser de dimensiones reducidas.

Debido a ello se realizaron pruebas de diseño de pads para capacitores, resistencias y diodos emisores de luz (LED's) en su empaquetado 0805.

Se diseñó y manufacturó una tarjeta de circuito impreso de un cupón de pads de los componentes mencionados, con el objetivo de verificar y validar los procesos de manufactura y la repetibilidad de definición de los pads.

A continuación, en la [Figura 3.4](#) inciso a) se muestra la tarjeta de circuito impreso en sustrato

de cobre del cupón de plantillas de empaquetado 0805 de montaje superficial.

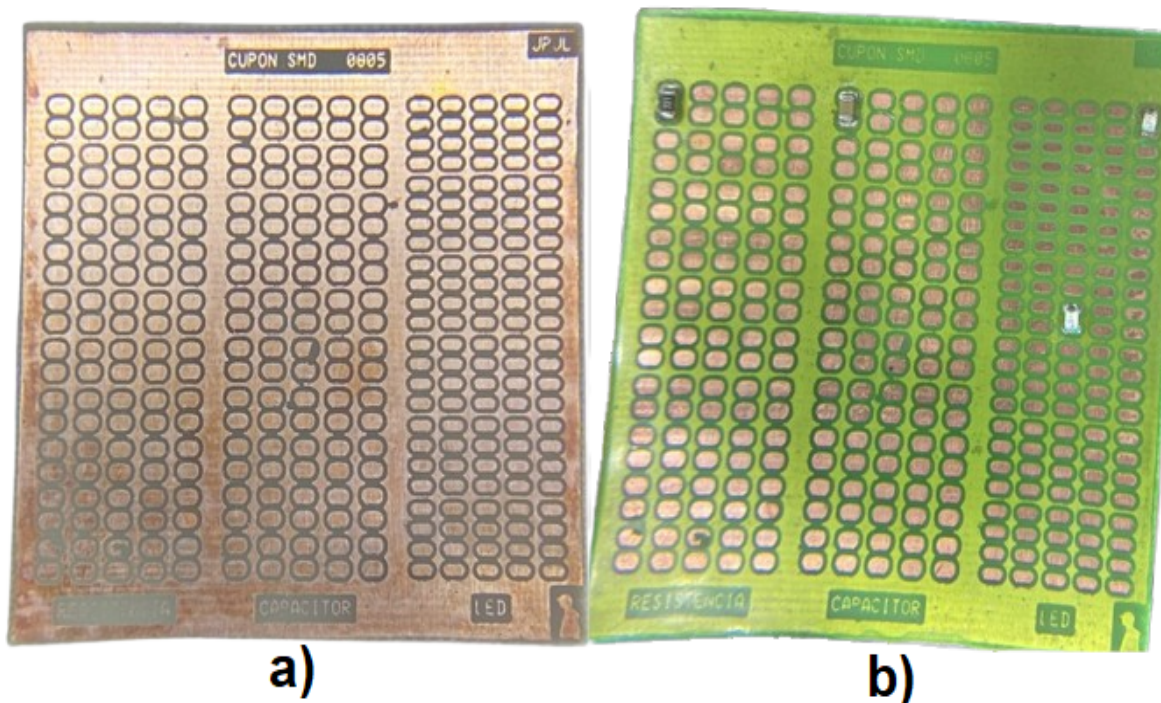


Figura 3.4: a) Tarjeta de circuito impreso en sustrato de cobre de Cupón de componentes de montaje superficial (SMD) de empaquetado 0805. b) Cupón componentes SMD 0805 con solder mask ultravioleta.

Con el objetivo de asegurar y practicar la soldadura de los componentes de montaje superficial, se propuso el recubrimiento de la tarjeta de circuito impreso, mostrado en el inciso b) de la [Figura 3.4](#). Consiste en una máscara de recubrimiento UV (Solder Mask) aplicada mediante un proceso de fotocurado, con el objetivo de evitar el deterioro u óxido del cobre de la tarjeta y facilitar la soldadura de los componentes.

Mediante el uso de dicha tarjeta de circuito impreso se practicó con mayor facilidad el proceso de soldadura de componentes de montaje superficial mediante la ayuda de la estación Baku601D con temperatura regulable y estaño de 0.2 mm de diámetro de la marca MacheNike. La temperatura a la que se configuró el cautín de la estación fue de 230 °C.

Una vez validado el funcionamiento de los componentes de montaje superficial y del proceso de soldadura de los componentes, se procedió con el último punto de la etapa 1 del desarrollo

del proyecto, que consiste en el diseño, la manufactura y el montaje del sistema embebido del sistema mínimo del microcontrolador ATmega328P-AU.

3.1.4 Diseño, manufactura y montaje del sistema embebido para microcontrolador ATmega328P-AU

El diseño de las tarjetas de circuito impreso realizadas en el proyecto de investigación fue realizado mediante el software de National Instruments Multisim y Ultiboard en su versión 14.0. Las tarjetas desarrolladas en el presente trabajo de investigación se basaron en la norma IPC-2221A.

Tomando como referencia el diagrama esquemático del circuito mínimo del microcontrolador en la [Figura 3.1](#), se procedió a realizar el diseño de la tarjeta de circuito impreso en Ultiboard 14.0 del sistema embebido para el microcontrolador ATmega328P. (Ver [Figura 3.5](#))

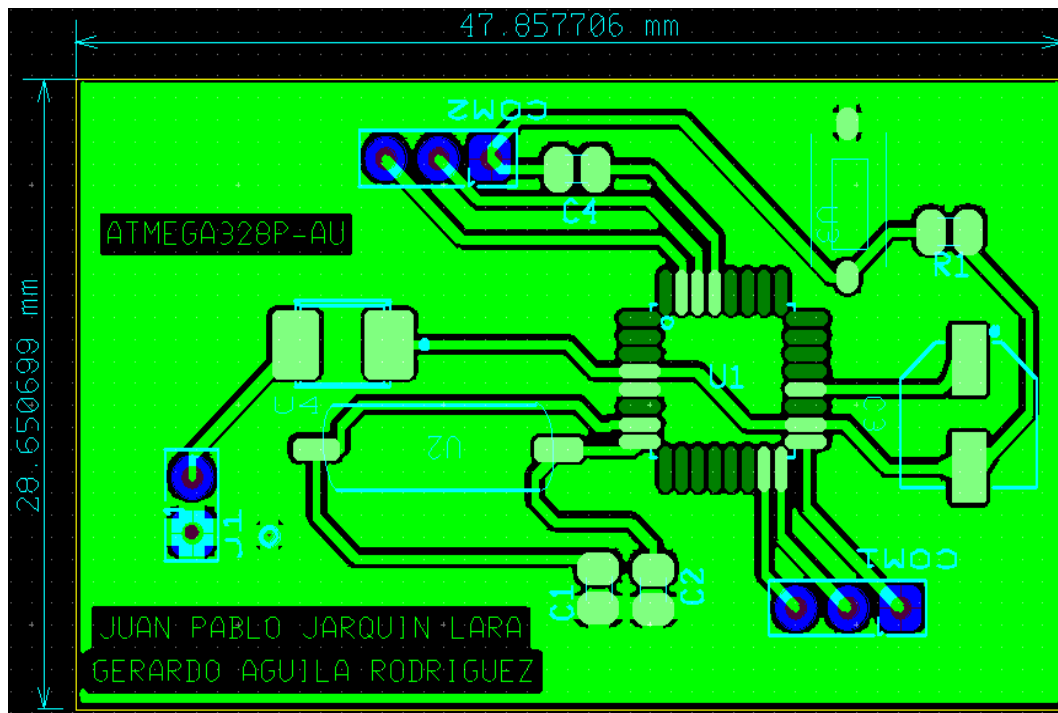


Figura 3.5: Diseño de tarjeta de circuito impreso para microcontrolador ATmega328P-AU en Ultiboard 14.0

Como se puede observar en la figura anterior, el diseño de la tarjeta de circuito impreso para el sistema mínimo de un microcontrolador obtuvo las dimensiones de un sustrato de cobre de

47.85 mm de largo por 28.65 mm de ancho.

Los pines de la tarjeta pertenecientes a la nomenclatura COM1 están conectados a los pines de comunicación ISP del microcontrolador (MISO, MOSI, SCK), con el propósito de permitir la programación del bootloader para un microcontrolador de fábrica, como paso previo a cargar un código.

Por otra parte, los pines de la tarjeta pertenecientes a la nomenclatura COM2 están conectados a los pines de transmisión y recepción del protocolo de comunicación UART para el microcontrolador (TX y RX), mientras que el pin restante está conectado al pin de Reset del microcontrolador, que pertenece al grupo de pines de comunicación ISP, con el propósito de poder reiniciar automáticamente el microcontrolador mediante una señal externa a la hora de cargar un nuevo código.

Posteriormente se realizó la manufactura de la tarjeta de circuito impreso mediante el mismo proceso de la tarjeta de circuito impreso del cupón de componentes de montaje superficial 0805 de la [Figura 3.4](#), es decir, por transferencia térmica de tóner y revelado mediante ataque químico de cloruro férrico.

Una vez desarrollada la habilidad para soldar los componentes de montaje superficial, se procedió a soldar aquellos que forman parte del sistema embebido para el microcontrolador ATmega328P-AU para integrar el sistema mínimo.

Con el sistema mínimo para el funcionamiento del microcontrolador desarrollado, es necesario abordar la integración de uno de los sensores a caracterizar y a integrar en la aplicación deseada. En este caso, se integra una metodología y el desarrollo experimental de un sistema embebido capaz de procesar señales capacitivas provenientes de un sensor basado en el principio de autocapacitancia.

3.2 METODOLOGÍA PARA EL DESARROLLO DE INTERFAZ CAPACITIVA (Etapa 2)

Una vez desarrollado el sistema embebido del microcontrolador de propósito general para la caracterización y configuración de los sensores y módulos del sistema, se procede a la Etapa

2: Análisis del principio de capacitancia mediante interfaces de sensores capacitivos. Esta fase tiene como objetivo comprender a profundidad el fenómeno físico de la capacitancia y establecer las bases para el diseño de interfaces precisas y confiables. Para ello, se inicia con la comprensión del fenómeno mediante el desarrollo de un circuito de descompensación de carga basado en el temporizador NE555, lo cual permite observar de manera práctica la variación de capacitancia y su relación con señales eléctricas medibles.

Posteriormente, se realiza la selección y configuración del sensor capacitivo, evaluando las distintas opciones tecnológicas disponibles y ajustando sus parámetros para optimizar la detección. Con base en estos resultados, se avanza al diseño de interfaces electrónicas que permitan una adecuada comunicación entre los sensores y el microcontrolador, asegurando estabilidad y precisión en la adquisición de datos. Finalmente, se lleva a cabo la manufactura, montaje y pruebas de los prototipos desarrollados, con el fin de validar su funcionamiento y garantizar que cumplan con los requerimientos técnicos establecidos. Esta etapa constituye un puente esencial entre la teoría y la aplicación práctica, permitiendo la integración de sensores capacitivos confiables en el sistema final.

3.2.1 Comprensión del fenómeno

Como primera instancia se debe comprender el fenómeno de capacitancia.

Para comprender el fenómeno, además de hacer una revisión a la teoría presentada en diferentes fuentes bibliográficas, se propusieron distintas geometrías para la realización de circuitos de prueba de sensores táctiles mediante el funcionamiento por descompensación de carga mediante un oscilador NE555 de la empresa National Instruments en su empaquetado SOIC de 8 pines de montaje superficial. Los componentes pasivos como resistencias y capacitores se establecieron en un empaquetado de montaje superficial 0805. El diámetro del área de cobre que realizaría la función de sensor táctil se propuso de un diámetro de 10 mm.

El diseño esquemático de este circuito se realizó en el software de National Instruments Multisim 14.0, como se puede ver a continuación en la [Figura 3.6](#).

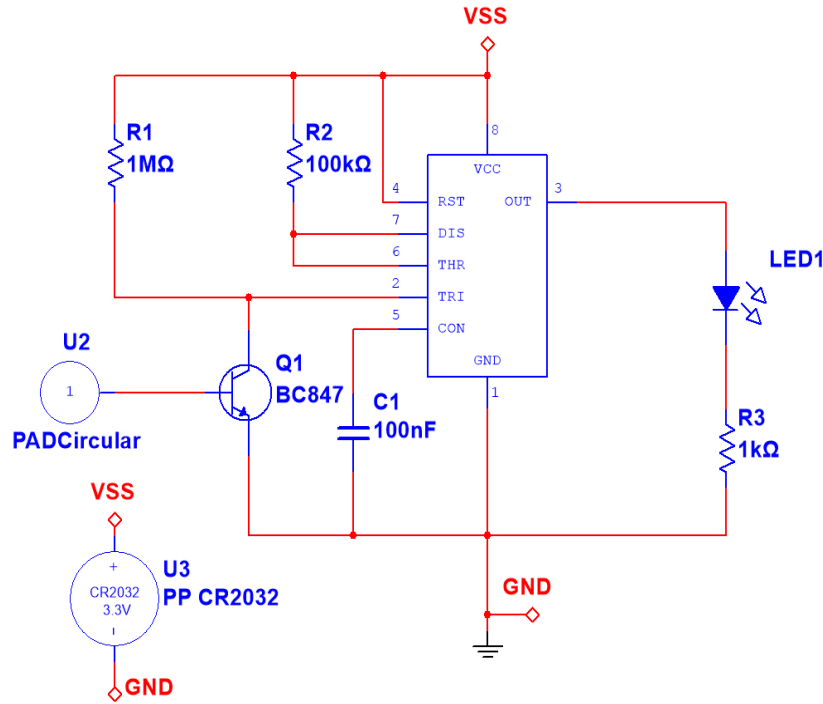


Figura 3.6: Diagrama esquemático de sensor capacitivo por descompensación de carga mediante oscilador NE555.

Una vez validado el diagrama esquemático del circuito y establecidos los componentes a utilizar, se procedió a realizar el diseño de la tarjeta de circuito impreso en el software Ultiboard 14.0 de National Instruments. Para el diseño se realizaron distintas propuestas, con el objetivo de experimentar diferentes geometrías de sensores y circuitos para evaluar la viabilidad eléctrica y mecánica de las tarjetas.

A continuación, en la [Figura 3.7](#) se muestra el diseño de la tarjeta de circuito para el circuito de sensor capacitivo basado en la descompensación de carga, diseñado en Ultiboard 14.0 de National Instruments.

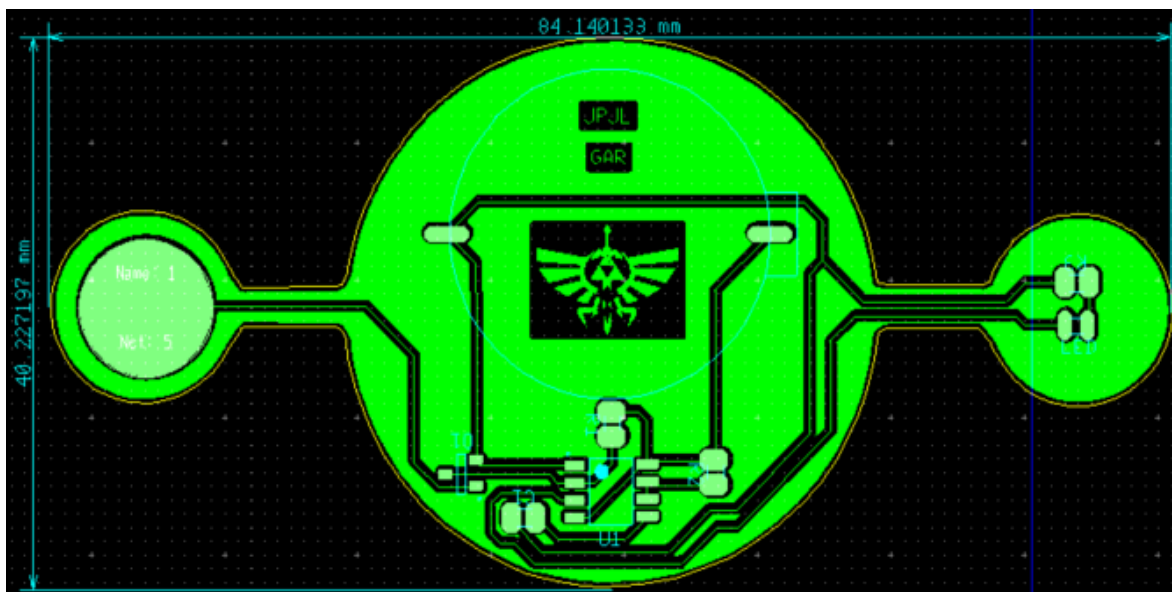


Figura 3.7: Diseño para manufactura de circuito de sensor capacitivo por descompensación de carga en Ultiboard 14.0.

Como se puede observar en la Figura anterior, el diseño se propuso con una geometría diferente a la típica geometría rectangular de una PCB. Esto se debe a que se pretende explorar los retos y dificultades tanto eléctricas como mecánicas que surgen a la hora de desarrollar una aplicación con una geometría diferente. La tarjeta tiene dimensiones de 84.14 mm de largo por 40.22 mm de ancho, siendo de 40 mm el diámetro de la geometría central y 14 mm de las geometrías externas.

Para realizar la manufactura de la tarjeta de circuito impreso se utilizó la técnica tradicional de transferencia térmica de tóner en el sustrato, con un ataque químico de cloruro férrico para el revelado del cobre.

Para ello se realizaron varias pruebas para encontrar mejorar la técnica y obtener un mejor resultado en la etapa de la remoción del cobre mediante el cloruro férrico, como se puede ver a continuación en la [Figura 3.8](#).



Figura 3.8: Pruebas de manufactura de tarjetas de circuito impreso de sensor capacitivo por descomensación de carga

Posteriormente, con ayuda de una mascarilla ultravioleta, se agregó una capa de protección a la tarjeta de circuito impreso para proteger el cobre de etapas de deterioro, como la generación de óxido. Posteriormente, se procedió a soldar aquellos componentes que forman parte del prototipo de sensor táctil por descompensación de carga. (Ver [Figura 3.9](#)).

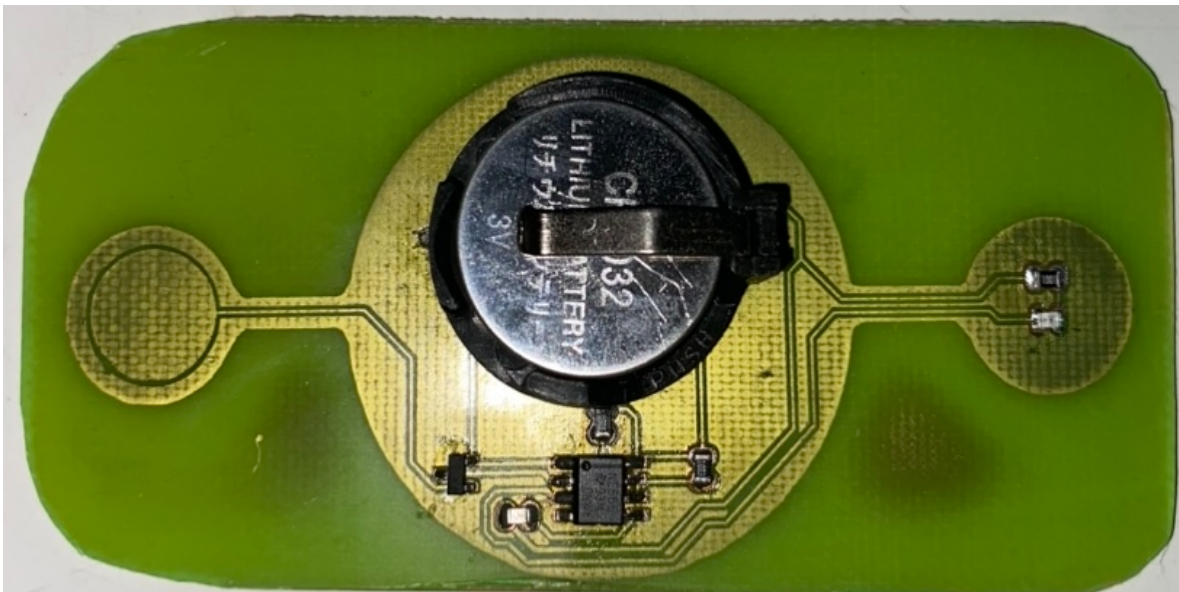


Figura 3.9: Prototipo de sensor táctil basado en descompensación de carga.

3.2.2 Análisis de diseño y selección de componentes

En esta subetapa se llevó a cabo la selección de los componentes principales para el sistema embebido encargado de la detección táctil mediante sensores capacitivos, considerando tanto el principio de funcionamiento del sensor como el procesamiento de las señales generadas. El objetivo principal fue elegir un dispositivo que permitiera realizar pruebas confiables, con bajo consumo energético y una arquitectura sencilla que favorezca la integración en un prototipo de asistencia para adultos mayores.

Se analizaron varios controladores táctiles de la familia AT42QT de Microchip Technology Inc., específicamente los modelos AT42QT1070, AT42QT2120, AT42QT2160 y AT42QT60248, los cuales difieren en número de canales, principio de funcionamiento y complejidad de implementación. El AT42QT1070 opera bajo el principio de autocapacitancia, en el que cada canal mide de manera independiente el cambio en la capacitancia de un electrodo con respecto a tierra. Este principio es ideal para aplicaciones simples donde no se requiere la identificación precisa de posición, sino únicamente la detección de toque. Por otro lado, modelos como el AT42QT2160 y el AT42QT60248 emplean capacitancia mutua, permitiendo el diseño de matrices que ofrecen detección bidimensional y funcionalidad multitáctil. Sin embargo, este tipo de sensores demanda un procesamiento más complejo, mayor consumo energético y un diseño más exigente en cuanto a la disposición de los electrodos y trazado de PCB.

Tras evaluar los requerimientos del sistema, se optó por el AT42QT1070, ya que ofrece un equilibrio óptimo entre simplicidad, bajo consumo de energía y cantidad suficiente de canales para las pruebas iniciales. Este dispositivo cuenta con 7 canales independientes, comunicación mediante I²C, y una configuración sencilla a través de pines de control, lo que permite implementar rápidamente un prototipo funcional. En contraste, el AT42QT2120 y el AT42QT2160, aunque ofrecen mayor número de canales (12 y 16 respectivamente), incrementan la complejidad y el consumo de energía, aspectos poco favorables para dispositivos portátiles y alimentados por batería. Finalmente, el AT42QT60248, aunque brinda una gran cantidad de entradas y funcionalidades avanzadas, excede por completo las necesidades del presente proyecto y representaría un sobre costo en términos de diseño y recursos.

De esta manera, la selección del AT42QT1070 como controlador táctil, junto con un diseño de electrodos basado en guías industriales, garantiza un sistema confiable, de bajo consumo y adecuado para la fase inicial de desarrollo de herramientas de asistencia mediante sensores capacitivos.

En cuanto a las geometrías de los electrodos, se consultó la guía de diseño de sensores capacitivos proporcionada por Microchip Technology Inc., la cual establece estándares para optimizar la sensibilidad, minimizar el ruido y garantizar una detección precisa. Estas geometrías definen el tamaño, forma y espaciado de los pads capacitivos, siendo fundamentales para asegurar un rendimiento estable del sistema. Entre las configuraciones más comunes se encuentran los electrodos circulares, cuadrados y en forma de anillo, cuya selección depende del área de contacto deseada y las limitaciones físicas del dispositivo. En la [Figura 3.10](#) se muestra un ejemplo de dichas geometrías estandarizadas, que sirvieron como referencia para el diseño de los pads empleados en este proyecto.

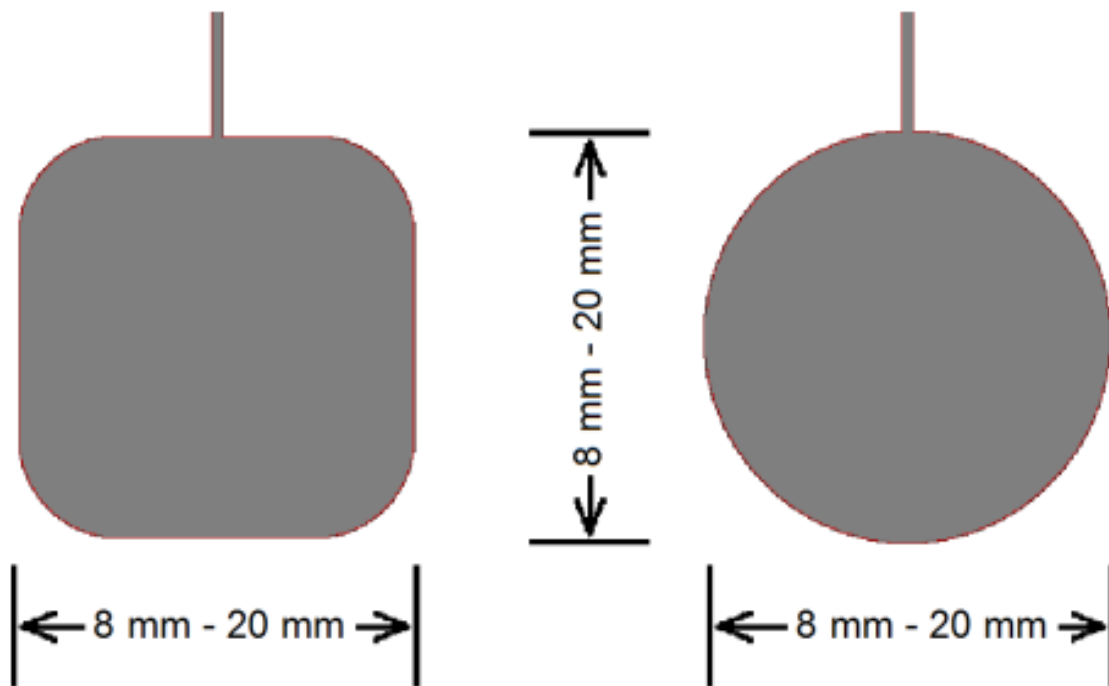


Figura 3.10: Estándar de geometrías para el diseño pads capacitivos. [31]

Como se puede analizar en la [Figura 3.10](#), el estándar de geometrías propuesto por Microchip

Technology Inc. nos recomienda una dimensión de 8 a 20 mm por lado para un sensor capacitivo o electrodo con geometría cuadrada redondeada, y una dimensión de 8 a 20 mm de diámetro para una geometría circular.

Apegándonos a [31], en la siguiente sección se realiza el diseño y la manufactura de las interfaces capacitivas a evaluar para un sistema embebido de procesamiento de señales capacitivas.

3.2.3 Diseño y desarrollo de interfaces

Para comenzar con el diseño de las interfaces capacitivas, como primera instancia se estableció el sensor a ocupar. En este caso, se seleccionó un sensor de la línea QT de Atmel, específicamente el sensor AT42QT1070, que es un sensor capacitivo de autocapacitancia que soporta 7 sensores o pads diferentes.

Para este punto se propusieron distintas geometrías de sensores capacitivos para evaluar su comportamiento mediante el sensor de la familia QT. A continuación, en la [Figura 3.11](#) se muestran diferentes geometrías de pads capacitivos diseñados y manufacturados.

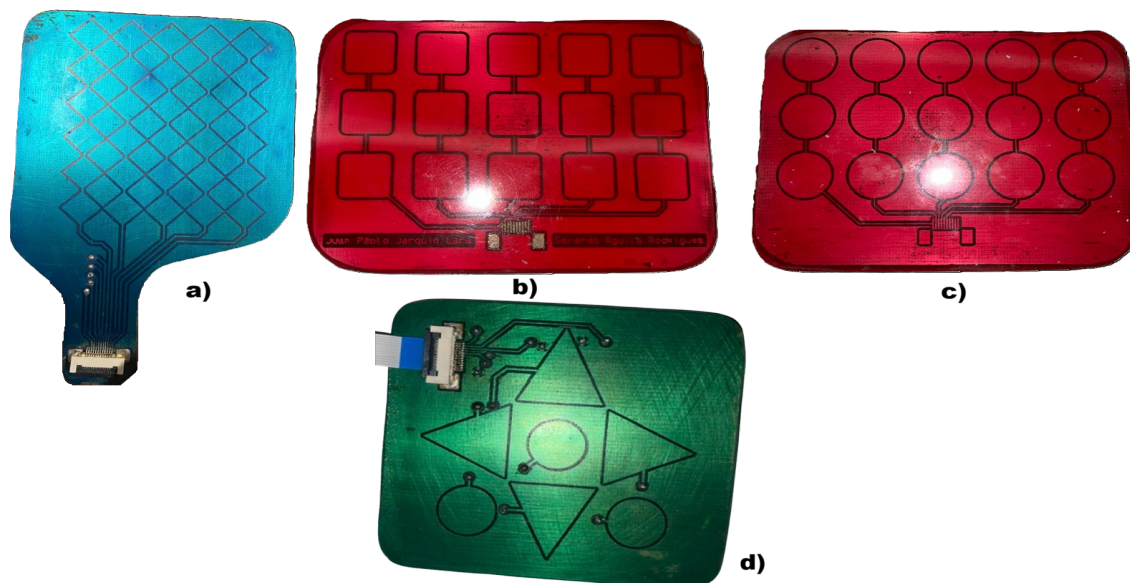


Figura 3.11: a) Interfaz capacitiva matricial para sensor de capacitancia mutua. b) Interfaz capacitiva con pads con forma cuadrada para autocapacitancia. c) Interfaz capacitiva con pads con forma circular para autocapacitancia. d) Interfaz de autocapacitancia para interconexión con interfaz visual de QTDesigner.

3.2.4 Manufactura, montaje y pruebas del sistema embebido de procesamiento de señales capacitivas

Finalmente, se realiza el diseño y la manufactura del sistema embebido para el sensor AT42QT1070 con bus de salida para el microcontrolador, el cual lleva una serie de componentes pasivos para funcionar correctamente. Para este paso, fue necesario recurrir a la hoja de datos del fabricante para obtener los valores y analizar las curvas características del comportamiento del sensor capacitivo.

A continuación, en la [Figura 3.12](#) se muestra la parte posterior de la interfaz capacitiva d) de la [Figura 3.11](#).

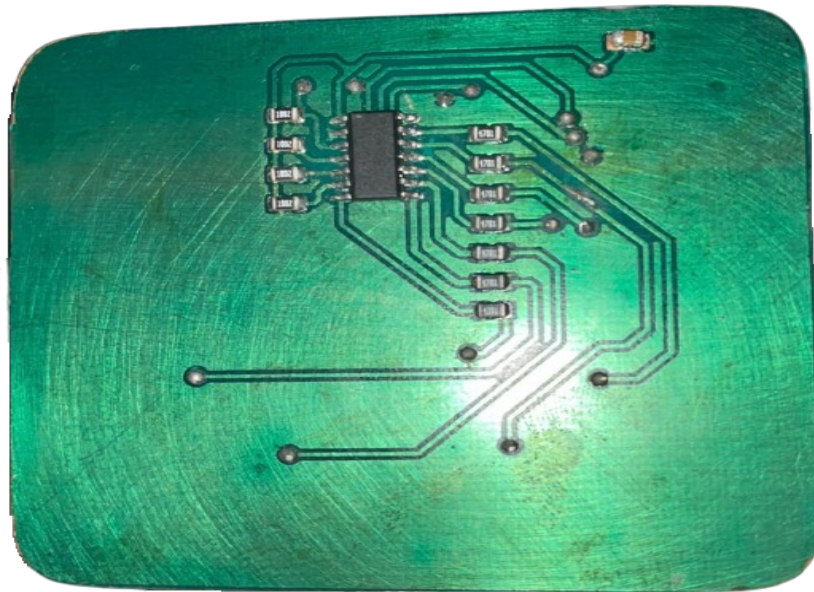


Figura 3.12: Sistema mínimo para el sensor capacitivo AT42QT1070.

A continuación, en la [Figura 3.13](#) se presenta una tarjeta de circuito impreso para el sensor capacitivo AT42QT1070 de manera independiente, desarrollada con el propósito de evaluar las interfaces de a), b) y c) de la [Figura 3.11](#).

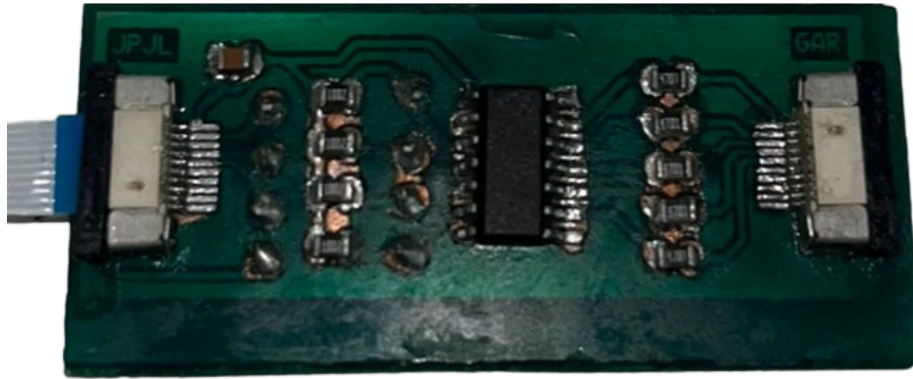


Figura 3.13: Sistema mínimo para el sensor capacitivo AT42QT1070 versión 2.

Para concluir con la etapa 2 del desarrollo del proyecto, con ayuda del diseño del sistema mínimo para el microcontrolador ATMega328P-AU elaborado en la etapa 1 ([Figura 3.5](#)), se adaptó dicho diseño para el procesamiento de señales capacitivas mediante un conector FPC de 12 pines de montaje superficial.

A continuación, en la [Figura 3.14](#) se muestra el diagrama esquemático de conexiones para el sistema embebido de procesamiento de señales capacitivas.

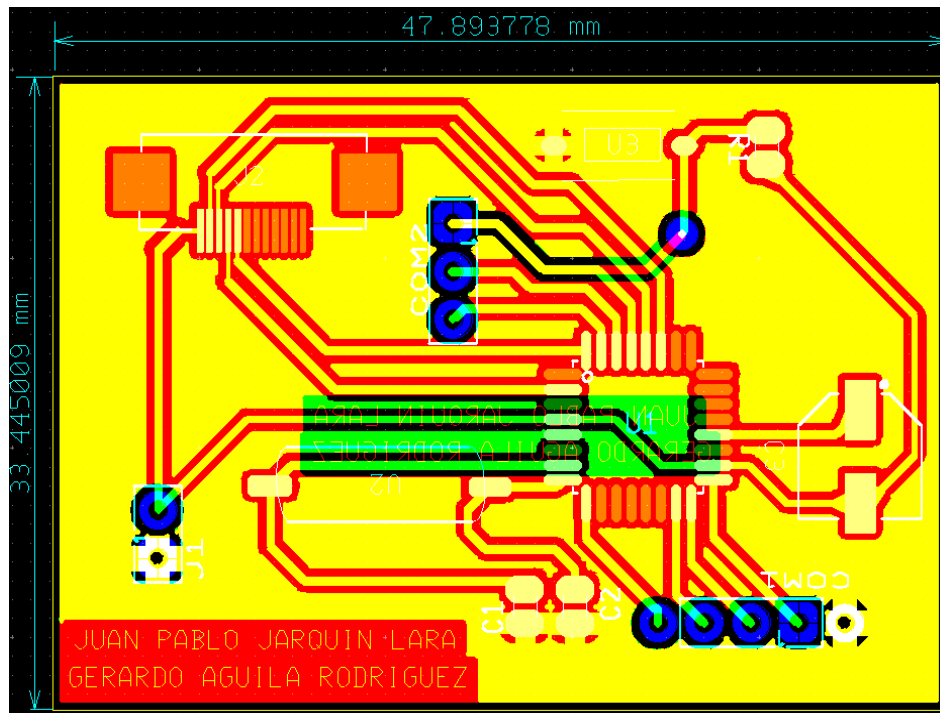


Figura 3.15: Diseño de tarjeta de circuito impreso de sistema embebido para procesamiento de señales capacitivas de AT42QT1070.

Finalmente se procede a realizar la manufactura del sistema embebido de procesamiento de señales capacitivas mediante las técnicas de transferencia de tóner y revelado por ataque químico de cloruro férrico, el resultado de la tarjeta de circuito impreso se presenta a continuación en la [Figura 3.16](#).

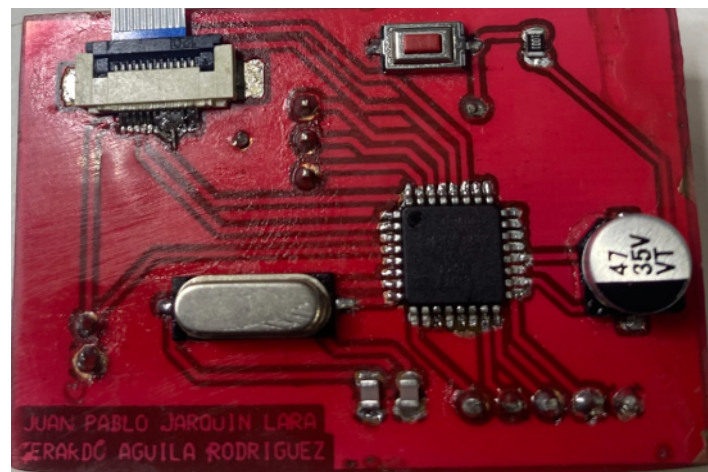


Figura 3.16: Sistema embebido para pruebas de sensor capacitivo AT42QT1070.

Para la caracterización del sensor capacitivo, fue necesario realizar pruebas de capacitancia, utilizando instrumentos de medición como capacitómetros o pinzas para medir la capacitancia de cada uno de los pads de las interfaces. De la misma manera, fue necesario hacer cambios en los registros del sensor para ajustar los umbrales de detección, con el objetivo de evitar falsos positivos en el proceso de detección.

Con el propósito de realizar pruebas y presentar un resultado visual mediante la interfaz capacitiva, se diseñó una interfaz en QTDesigner que representa los sensores capacitivos del inciso d) de la Figura 22, programada en Python para leer las señales provenientes del microcontrolador del sistema embebido mediante las librerías de PySerial. Dicha interfaz se presenta como uno de los resultados en el Capítulo 4.

Una vez validado el sistema embebido para el procesamiento de señales capacitivas, se continúa con la etapa 3 del desarrollo del trabajo de investigación, que consiste en validar los módulos de comunicación y alerta.

3.3 METODOLOGÍA PARA EL DESARROLLO DE UN SISTEMA EMBEBIDO DE COMUNICACIÓN GSM (Etapa 3)

Una vez analizados los principios de funcionamiento de los sensores capacitivos y definida la interfaz necesaria para la detección e interpretación de eventos físicos, la siguiente fase se centra en la integración de un sistema de comunicación que permita transmitir la información obtenida hacia un medio externo de manera confiable y en tiempo real. Esta etapa tiene como objetivo el diseño y desarrollo de un sistema embebido basado en un módulo de comunicación GSM, que funcione como enlace entre el dispositivo de sensado y el usuario final o una plataforma de monitoreo.

En esta fase se seleccionarán y configurarán los componentes electrónicos adecuados, considerando factores como consumo energético, tamaño, compatibilidad con la arquitectura del sistema y estabilidad en la transmisión de datos. Asimismo, se establecerá la lógica de control necesaria para gestionar la comunicación entre el microcontrolador principal y el módulo GSM, contemplando la recepción de datos provenientes de los sensores, su procesamiento y la

ejecución de acciones como el envío de alertas mediante llamadas o mensajes de texto.

El desarrollo de este sistema no solo busca garantizar la autonomía del dispositivo, sino también asegurar que la comunicación se realice de manera eficiente y robusta. Esta etapa es clave para sentar las bases de la implementación final del prototipo funcional, orientado a la asistencia y monitoreo de adultos mayores, cumpliendo con los objetivos planteados en el proyecto.

3.3.1 Revisión y análisis de las características técnicas del módulo de comunicación GSM

Para continuar con la etapa 3 del desarrollo experimental del presente proyecto de investigación, el primer paso a seguir es la revisión exhaustiva de la hoja de datos del módulo seleccionado mediante la TABLA DE MÓDULOS (CAP1), que propone como uno de los módulos ideales para el propósito del trabajo, el módulo SIM800L, el cual es un módulo de comunicación GSM/GPRS de cuádruple banda (850/900/1800/1900 MHz) que permite la transmisión y recepción de datos, así como la realización de llamadas y el envío de mensajes SMS mediante comandos AT CITAR ANEXO. Su diseño compacto y bajo consumo energético lo hacen ideal para aplicaciones embebidas portátiles. Opera con un voltaje de alimentación de 3.4 V a 4.4 V y se comunica con el microcontrolador a través de una interfaz UART, lo que facilita su integración en sistemas de monitoreo y control remoto. CITAR ANEXO HOJA DE DATOS

Las siglas GSM, por su nombre en inglés Global System for Mobile Communications, hacen referencia al sistema global para las comunicaciones móviles 2G desarrollado por el Instituto Europeo de Normas de Telecomunicaciones (ETSI) que es libre de regalías y de telefonía móvil digital.

A continuación, en la [Figura 3.17](#) se muestra la configuración de pines del módulo de comunicación GSM SIM800L de la empresa SimCom.

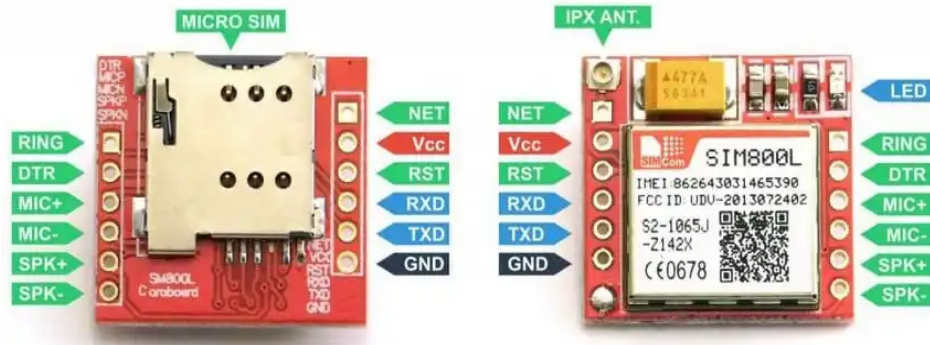


Figura 3.17: Distribución de pines de conexión del módulo SIM800L CITAR IMAGEN

Con una revisión detallada a la hoja de datos del fabricante y a la operación de los denominados Comandos AT, se elaboró un código sencillo para configurar el módulo SIM800L en modo de transmisión de mensajes de texto SMS. Para ello fue necesaria la adquisición de una tarjeta SIM compatible con 2G. Actualmente, en México, la mayoría de compañías aún cuentan con servicio de red de telefonía móvil 2G.

3.3.2 Selección de componentes y pruebas preliminares

Con el propósito de probar los comandos AT y verificar el funcionamiento del módulo de comunicación, se propusieron una serie de códigos sencillos mediante el microcontrolador ATmega328P para enviar comandos AT al módulo y configurarlo para enviar un mensaje SMS a un número configurado en el código.

A continuación se muestra en la [Figura 3.18](#), la pantalla del dispositivo al recibir un mensaje SMS por parte del módulo al subir el código para enviar un mensaje de texto mediante los comandos AT. De la misma manera se muestra la conexión del módulo de comunicación GSM en un Protoboard con el objetivo de verificar las conexiones y evaluar la funcionalidad del mismo.

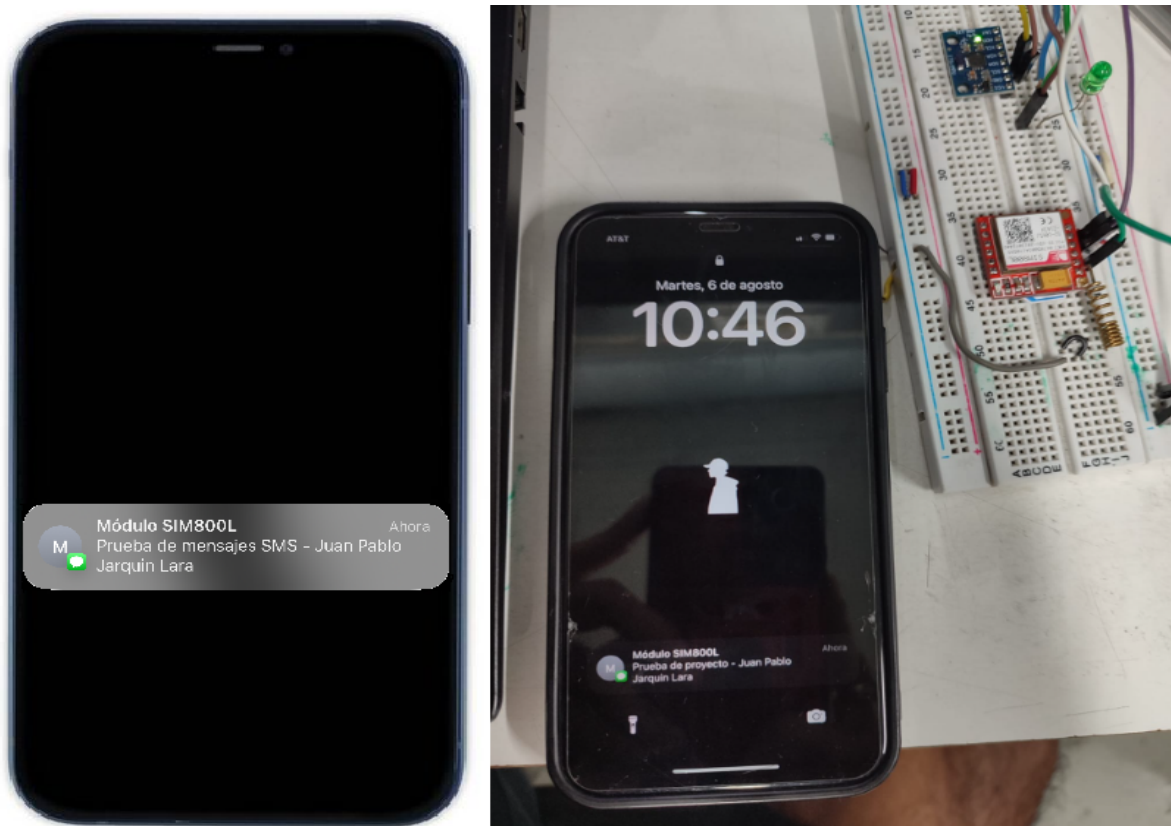


Figura 3.18: Prueba de módulo SIM800L para enviar mensajes de texto SMS mediante microcontrolador ATmega328P

Así mismo, se elaboró un código sencillo para realizar una llamada telefónica al número predeterminado, como se puede ver en la [Figura 3.19](#).

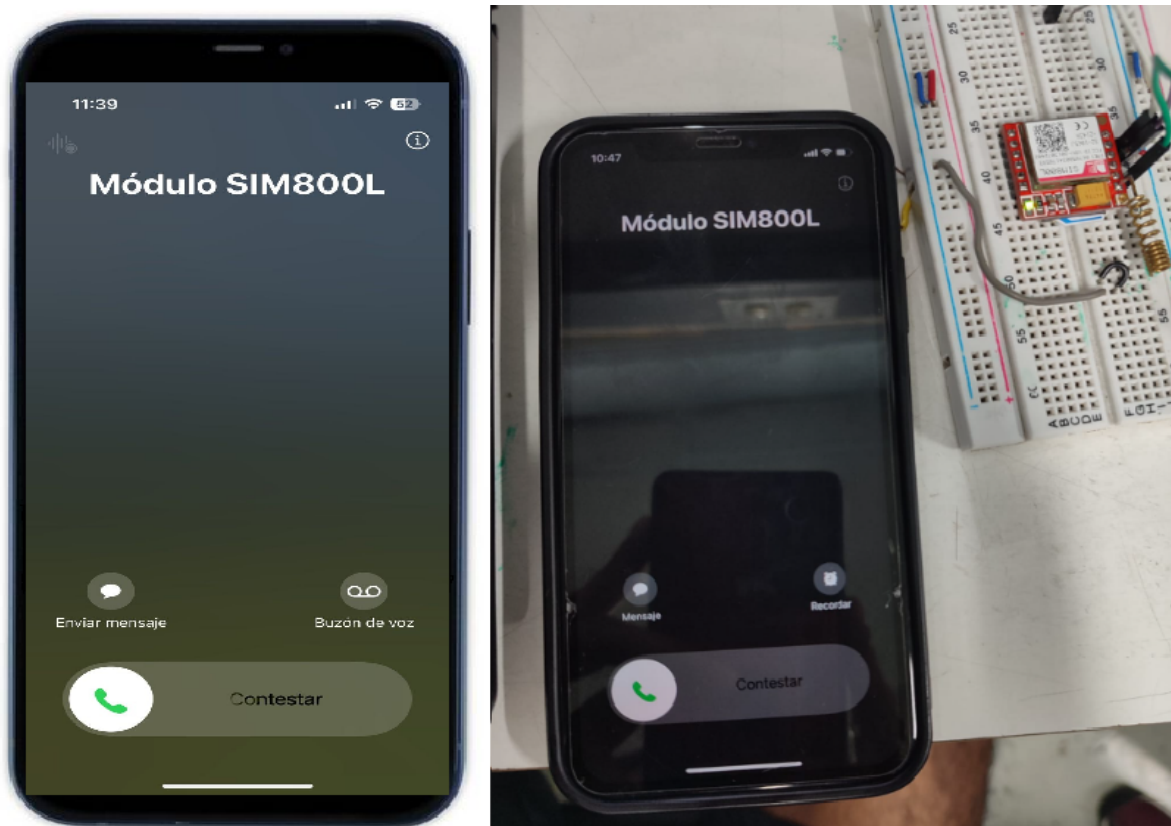


Figura 3.19: Prueba experimental de módulo SIM800L para llamadas telefónicas mediante microcontrolador ATmega328P.

Se realizaron de la misma manera pruebas de recepción de mensajes de texto para controlar el encendido y apagado de un parpadeo de un diodo emisor de luz (LED) como se puede ver a continuación en la [Figura 3.20](#).

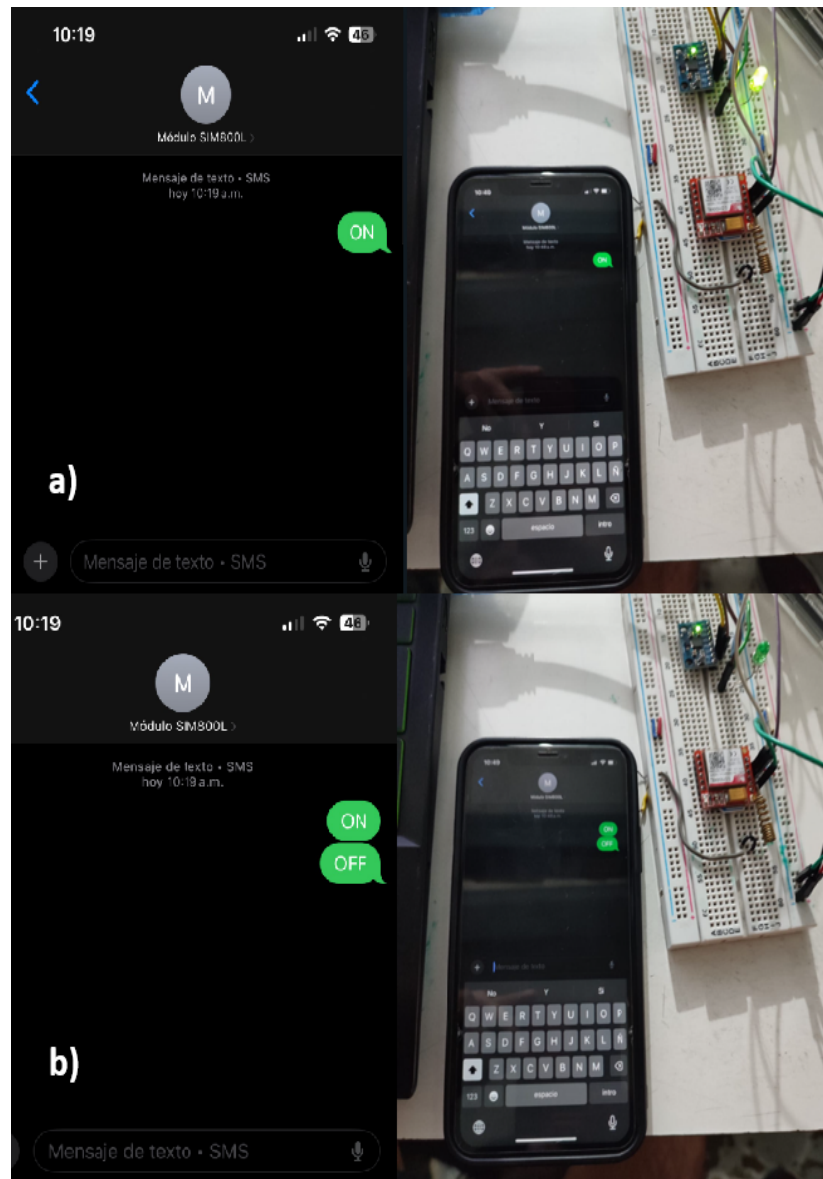


Figura 3.20: Prueba experimental de módulo SIM800L para recepción de mensajes de texto mediante microcontrolador ATmega328P.a) Envío de la palabra ON vía SMS al módulo para encender LED. b) Envío de la palabra OFF vía SMS al módulo para apagar LED.

Finalmente se realizó la prueba del módulo de comunicación junto con el sensor de movimiento inercial MPU6050, siendo que al detectar un cambio en la posición del acelerómetro con giroscopio, se enviaría un mensaje de texto al número telefónico predeterminado, como se puede ver a continuación en la [Figura 3.21](#).

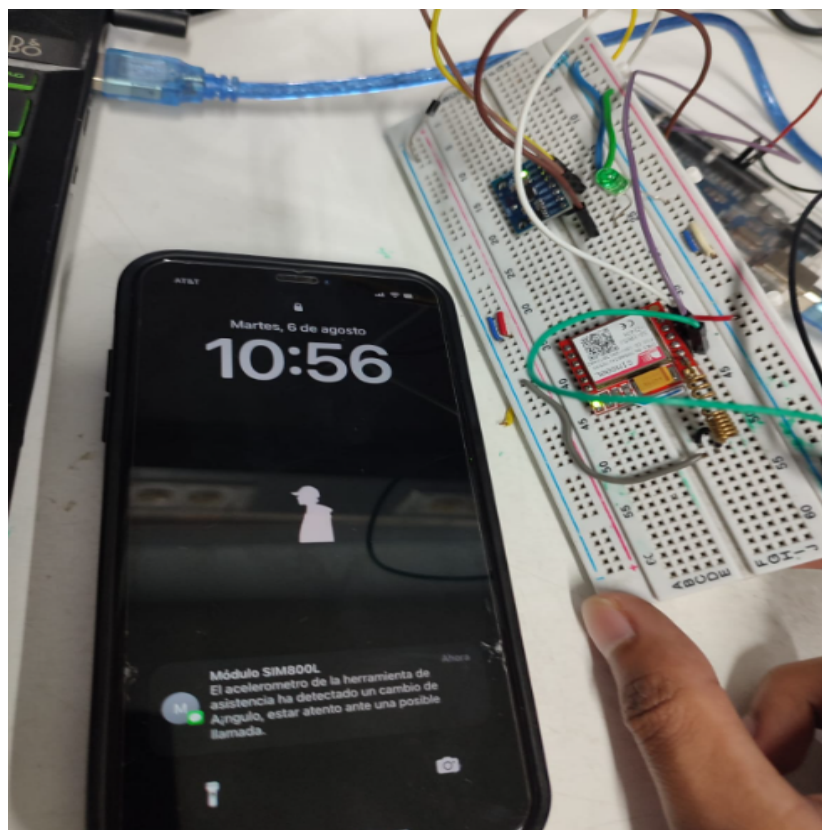


Figura 3.21: Prueba experimental de sensor de movimiento inercial integrado con módulo de comunicación GSM.

3.3.3 Diseño y ajustes de código mediante comandos AT

Una vez comprobado el funcionamiento del sensor de movimiento inercial para enviar un mensaje de texto mediante el módulo de comunicación GSM, se comenzó por diseñar los prototipos de prueba de los sistemas embebidos para cada uno de los módulos, comenzando por la integración de los dispositivos, incluyendo los periféricos de audio, como micrófonos y altavoces, necesarios para la realización y recepción de llamadas telefónicas a través del módulo de comunicación GSM.

Para ello fue necesario una vez más analizar la hoja de datos del fabricante y observar sus recomendaciones de conexiones, los tipos de salida de audio del módulo GSM y su configuración de canales de salida para así evaluar la adición de un posible amplificador de audio.

Con el objetivo de aumentar la ganancia de la señal se utilizó un amplificador de audio PAM8403. El PAM8403 es un amplificador de audio digital Clase D de bajo consumo y alta eficiencia, diseñado para aplicaciones portátiles. Proporciona una salida estéreo de hasta 3 W por canal a 4 ohm con una alimentación de 5 V, sin requerir disipador de calor debido a su bajo nivel de generación térmica. Este módulo amplifica la señal de audio proveniente del SIM800L, permitiendo obtener un volumen adecuado para la comunicación por voz a través de altavoces de pequeño tamaño.

A continuación, en la Figura [Figura 3.22](#), se muestra una prueba en Protoboard del módulo de amplificación de audio PAM8403 en Protoboard junto con el módulo de comunicación GSM SIM800L, con sus periféricos de audio (micrófono y bocinas) integrados para evaluar la calidad de una llamada telefónica programada mediante el código del microcontrolador ATmega328P.

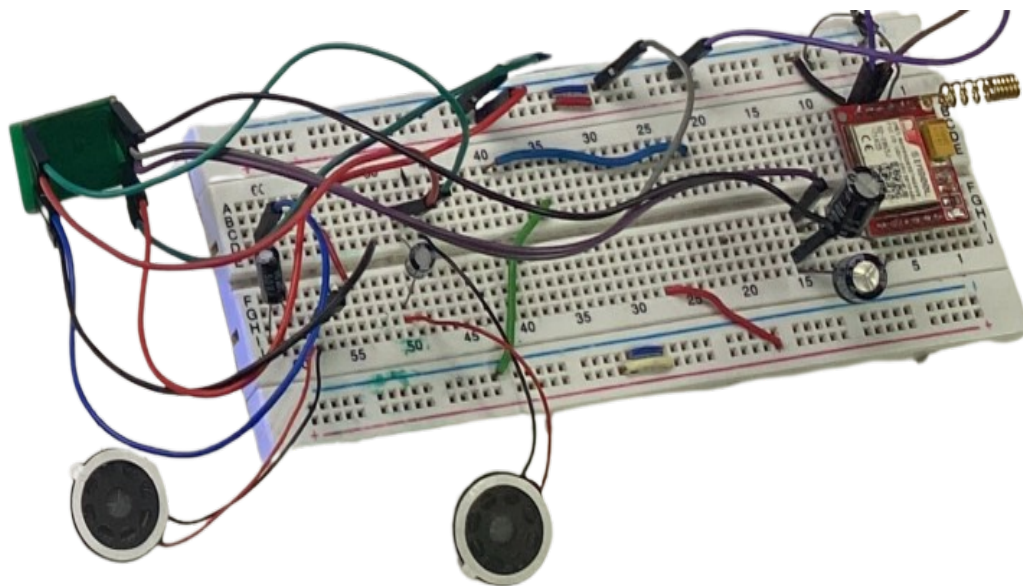


Figura 3.22: Prueba experimental de periféricos de audio integrados con módulo de comunicación GSM.

Una vez validada la calidad de la llamada mediante los componentes pasivos y los periféricos de audio, se procede a realizar el diseño de la tarjeta de circuito impreso para integrar dichas conexiones y reducir impedancias parásitas a causa de los cables de conexión utilizados en las

pruebas de Protoboard, como se puede observar en la Figura anterior.

Para lograr un acoplamiento adecuado para la señal de entrada del micrófono y salida por las bocinas del módulo GSM durante las llamadas, fue necesario realizar pruebas de calibración mediante los componentes pasivos y un osciloscopio, analizando a detalle las frecuencias registradas al monitorear la voz de un usuario para agregar filtros con el objetivo de mejorar la calidad de la señal.

A continuación, en la [Tabla 3.1](#) se muestran las tablas de frecuencias obtenidas en las pruebas de filtros para el acondicionamiento de la señal de audio del módulo GSM.

Tabla 3.1: Configuraciones de filtros RC para las salidas de audio del módulo GSM

Frecuencia de corte	Resistencia	Capacitor cerámico	Capacitor electrolítico
1.59 kHz	1 k Ω	0.1 μ F	10 μ F
8.84 kHz	180 Ω	0.1 μ F	10 μ F
10.61 kHz	150 Ω	0.1 μ F	10 μ F
15.92 kHz	100 Ω	0.1 μ F	10 μ F

La voz humana se concentra en el rango de 300 Hz a 3.4 kHz, por lo que se diseñaron filtros RC con distintas frecuencias de corte (1.59 kHz, 8.84 kHz, 10.61 kHz y 15.92 kHz) conectados a las líneas SPK+ y SPK- del módulo GSM.

Estos filtros permiten analizar la respuesta del sistema ante diferentes niveles de atenuación y asegurar una reproducción clara de la voz durante las llamadas. Además, se registró la polarización de cada filtro para garantizar la correcta conexión de los capacitores electrolíticos y la fidelidad del audio en las bocinas.

La configuración final seleccionada para el acondicionamiento de la señal de audio fue a una frecuencia de corte de 8.84 kHz, permitiendo una transmisión clara de la voz humana en la llamada telefónica a través de las bocinas.

3.3.4 Manufactura, montaje y pruebas del sistema embebido para comunicación GSM

A continuación, en la [Figura 3.23](#) se muestra el diseño del sistema embebido para la gestión de llamadas y mensajería del módulo de comunicación GSM.

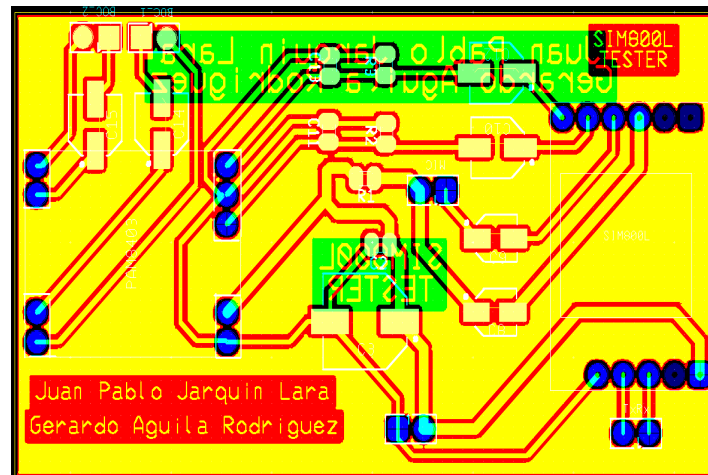


Figura 3.23: Prueba experimental de periféricos de audio integrados con módulo de comunicación GSM.

Para continuar con la manufactura de la tarjeta de circuito impreso del sistema embebido, se utilizaron las técnicas de las etapas anteriores, en las cuales se toman medidas de seguridad para garantizar el éxito del proceso de manufactura mediante el uso de equipo de seguridad, como lo son lentes de seguridad, guantes y mascarillas para el proceso de ataque químico al sustrato de cobre.

Se muestra a continuación el resultado del sistema embebido para comunicación GSM en la [Figura 3.24](#)



Figura 3.24: Tarjeta de circuito impreso de sistema embebido para comunicación GSM.

Como se puede observar en la [Figura 3.24](#), la tarjeta del sistema embebido para la gestión de comunicaciones GSM cuenta con los componentes pasivos necesarios para la comunicación vía llamada telefónica. En este caso, cuenta con un micrófono electret para monitorear la voz del usuario, mientras que las bocinas reproducen la señal de audio entrante, transmitida por el módulo GSM.

Una vez desarrollado el sistema embebido para la comunicación GSM, se procede a la etapa 4 de la metodología correspondiente al desarrollo de un sistema embebido para la Unidad de Medición Inercial (IMU) propuesta.

3.4 METODOLOGÍA PARA EL DESARROLLO DE SISTEMA EMBEBIDO PARA REGISTRO DE DATOS INERCIALES (Etapa 4)

La etapa 4 de la metodología se centra en el desarrollo del sistema embebido encargado de la adquisición, gestión y transmisión de los datos provenientes de la Unidad de Medida Inercial. En esta fase se define la arquitectura electrónica, la configuración del sensor, el manejo de interrupciones y los protocolos de comunicación necesarios para asegurar un registro confiable, consistente y sincronizado de las señales inerciales. Además, se establecen los procedimientos de captura y almacenamiento que permitirán conformar la base de datos utilizada durante el entrenamiento y validación del modelo neuromórfico. Esta etapa constituye el vínculo

operativo entre el entorno físico y el procesamiento algorítmico posterior, garantizando que la información registrada mantenga la integridad y la calidad requeridas para el análisis y la inferencia en las fases subsecuentes.

3.4.1 Análisis preliminar de especificaciones de la Unidad de Medición Inercial (IMU)

Para la adquisición de datos inerciales en el sistema embebido, se seleccionó el MPU6050, un sensor de movimiento de seis grados de libertad (6DoF) que integra un acelerómetro triaxial y un giróscopo triaxial en un solo encapsulado. Este dispositivo fue elegido por su alta integración, bajo costo, tamaño compacto y facilidad de comunicación, lo que lo hace particularmente adecuado para aplicaciones portátiles como la detección de caídas.

A diferencia de soluciones más avanzadas como el MPU9250 o el LSM9DS1, que también incluyen magnetómetro y mayor capacidad de procesamiento interno, el MPU6050 ofrece un conjunto suficiente de mediciones (aceleración y velocidad angular) para detectar eventos dinámicos como caídas, manteniendo una baja complejidad y menor demanda energética. Esta característica es crucial para sistemas embebidos donde el consumo de energía y la eficiencia computacional son factores clave.

El sensor se comunica vía interfaz I2C, lo que permite una integración sencilla con microcontroladores como el ATmega328P sin requerir hardware adicional. Además, cuenta con una unidad de procesamiento digital de movimiento (DMP) interna, que puede ser aprovechada para realizar filtrado y fusión de sensores, aunque en este trabajo se optó por utilizar directamente los datos crudos del acelerómetro para un análisis más fino de las señales asociadas a caídas.

En comparación con otros sensores de aceleración de uso común como el de series como ADXL o el MMA8451, el MPU6050 proporciona una frecuencia de muestreo más alta y una resolución mayor (16 bits), además de incluir el giróscopo en el mismo encapsulado, lo que reduce el número de componentes y el tamaño del sistema.

Estas características lo convierten en una opción adecuada tanto desde el punto de vista técnico como económico para el prototipado y evaluación de eventos de caída en sistemas embebidos

de bajo consumo.

El sensor MPU6050 permite la configuración mediante registros de la sensibilidad tanto para su acelerómetro como para el giroscopio integrado en el mismo encapsulado.

A continuación en la [Tabla 3.2](#) y [Tabla 3.3](#) se muestran los rangos de medición del acelerómetro y giroscopio, así como sus valores de sensibilidad, respectivamente.

Tabla 3.2: Sensibilidad del acelerómetro del MPU6050.

Rango de Medición (g)	Sensibilidad (LSB/g)
$\pm 2g$	16,384
$\pm 4g$	8,192
$\pm 8g$	4,096
$\pm 16g$	2,048

Tabla 3.3: Sensibilidad del giroscopio del MPU6050.

Rango de Medición ($^{\circ}/s$)	Sensibilidad (LSB/ $^{\circ}/s$)
± 250	131
± 500	65.5
± 1000	32.8
± 2000	16.4

3.4.2 Configuración de la Unidad de Medición Inercial

Para realizar pruebas del sensor se realizó un código de Arduino en el cual imprimimos mediante el monitor serial, los valores de los registros del sensor para analizar el comportamiento de los ejes X, Y y Z.

Los resultados en el monitor serial de la medición del registro del sensor se da en LSB (Least Significant Bits) o Bits Menos Significativos por segundo (LSB/s). Para obtener un valor con mayor interpretación, se realizó una conversión en el código, conociendo que el sensor

MPU6050 tiene una sensibilidad de 131 LSB por °/s en su configuración de fábrica, que es a ± 250 °/s .

Por lo tanto, el valor entregado en LSB/s lo dividimos entre 131, que es el valor de la sensibilidad para obtener un valor en grados por segundo (°/s), como se puede ver a continuación en la [Figura 3.25](#).

3.4.3 Pruebas preliminares y ajuste del código para monitoreo

```
int16_t gyroX, gyroY, gyroZ;
mpu.getRotation(&gyroX, &gyroY, &gyroZ);

float angleX = gyroX; / 131.0;
float angleY = gyroY; / 131.0;
float angleZ = gyroZ; / 131.0;

Serial.print("X: "); Serial.print(angleX);
Serial.print(" Y: "); Serial.print(angleY);
Serial.print(" Z: "); Serial.println(angleZ);
```

Figura 3.25: Código para analizar el comportamiento del giroscopio del sensor MPU6050 en °/s configurado a ± 250 °/s.

3.4.4 Diseño, manufactura y validación del sistema embebido de monitoreo inercial

Para realizar el diseño de la tarjeta de circuito impreso para el sensor de movimiento inercial, se comenzó por validar el diagrama esquemático de conexiones del sistema embebido, como se puede ver en la [Figura 3.26](#).

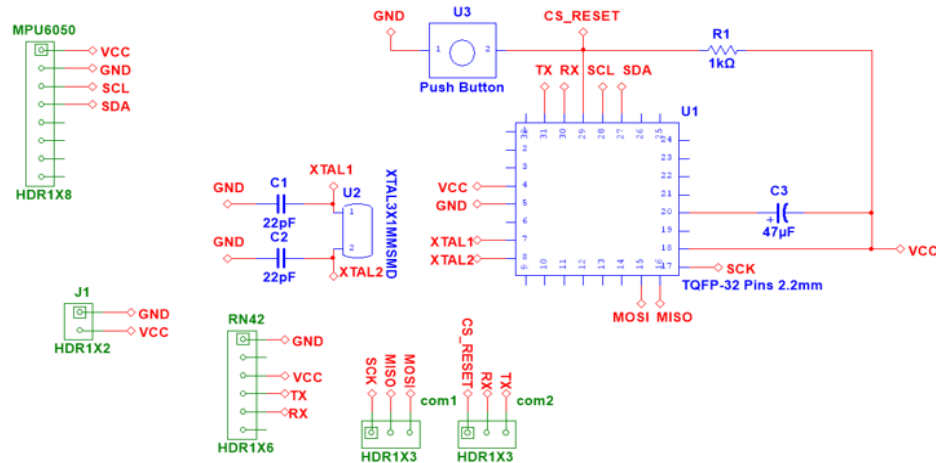


Figura 3.26: Diagrama esquemático para sistema embebido para pruebas de sensor de movimiento inercial.

Una vez validado y comprobado el diagrama esquemático de las conexiones del sensor de movimiento inercial, del módulo de comunicación Bluetooth y del sistema mínimo para el microcontrolador, se procedió a diseñar la tarjeta de circuito impreso en el software Ultiboard de National Instruments. (Ver [Figura 3.27](#))

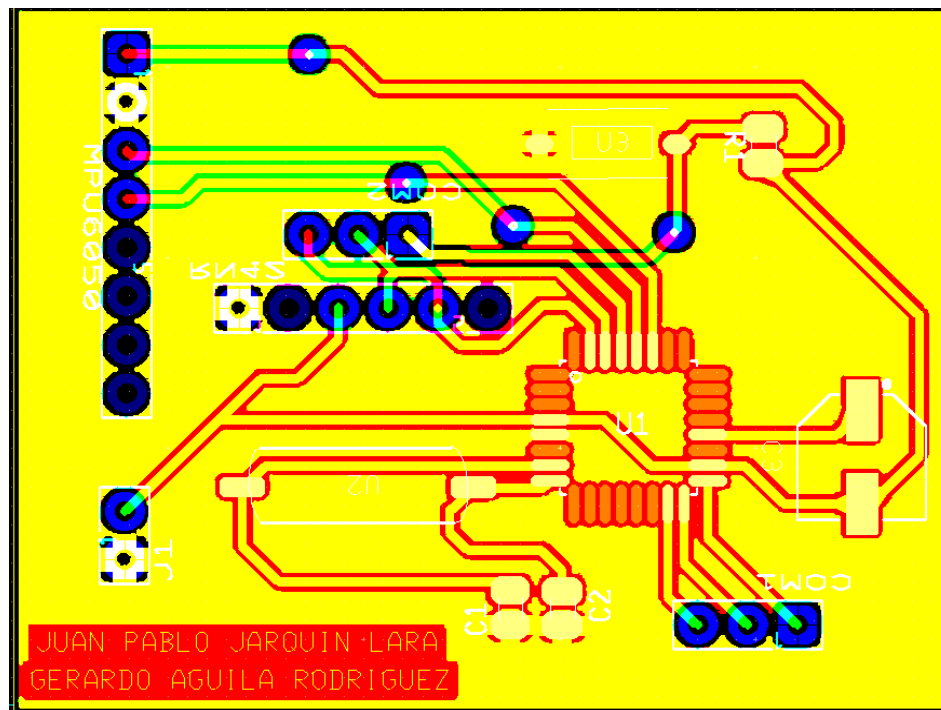


Figura 3.27: Diseño de sistema embebido para pruebas de sensor de movimiento inercial.

Para la manufactura de la tarjeta de circuito impreso, se utilizó, como en los demás casos, la técnica de transferencia térmica de tóner y el revelado mediante cloruro férrico, como se puede observar a continuación en la [Figura 3.28](#).

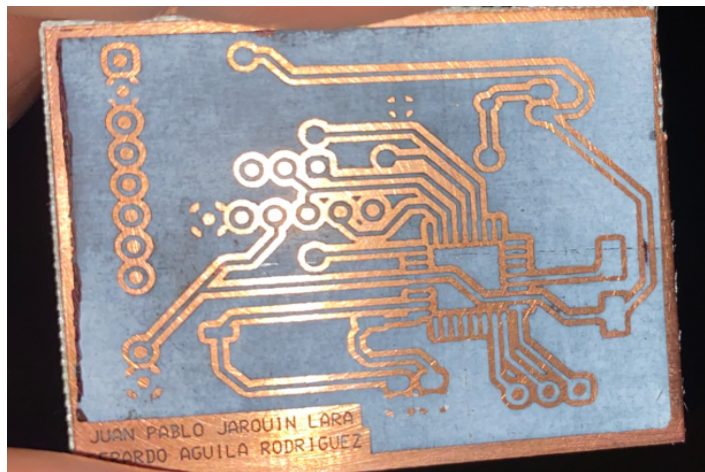


Figura 3.28: Revelado de tarjeta de circuito impreso del sistema embebido para pruebas de sensor de movimiento inercial.

Posteriormente, se agregó una máscara de pintura ultravioleta como capa de protección para el cobre del circuito y se revelaron los pads para poder realizar la soldadura de los componentes del sistema. A continuación, en la [Figura 3.29](#) se muestra el resultado preliminar del sistema embebido para el registro inercial.

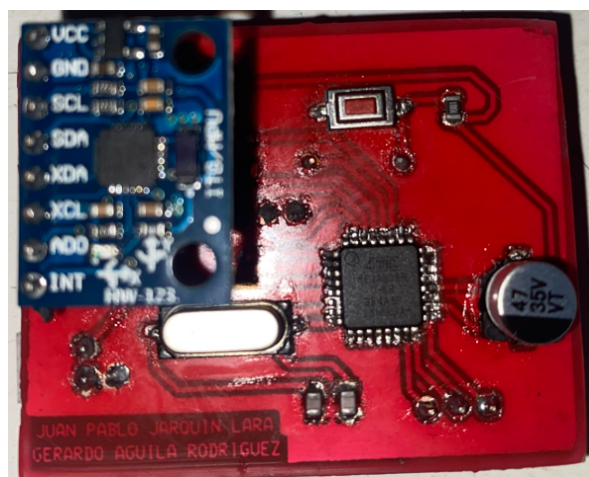


Figura 3.29: Sistema embebido para pruebas de sensor de movimiento inercial.

Se realizó una serie de pruebas preliminares para determinar la configuración óptima para obtener datos cruciales para la determinación de patrones de cada uno de los movimientos estipulados, descartando posibles saturaciones del sensor para tener la mayor confiabilidad posible en nuestra aplicación de Inteligencia Artificial para la identificación de patrones.

El procedimiento experimental para esta fase del proyecto consistió en diseñar una estructura para colocar el sistema embebido de monitoreo inercial sobre una base y dejarlo en caída libre desde una altura determinada, comparando las aceleraciones alcanzadas en una caída libre con amortiguación en la base contra un impacto directo con la superficie.

A continuación, en la [Figura 3.30](#) se presenta un esquema general de la estructura y el procedimiento experimental para la fase de pruebas del sistema embebido de registro de datos inerciales.

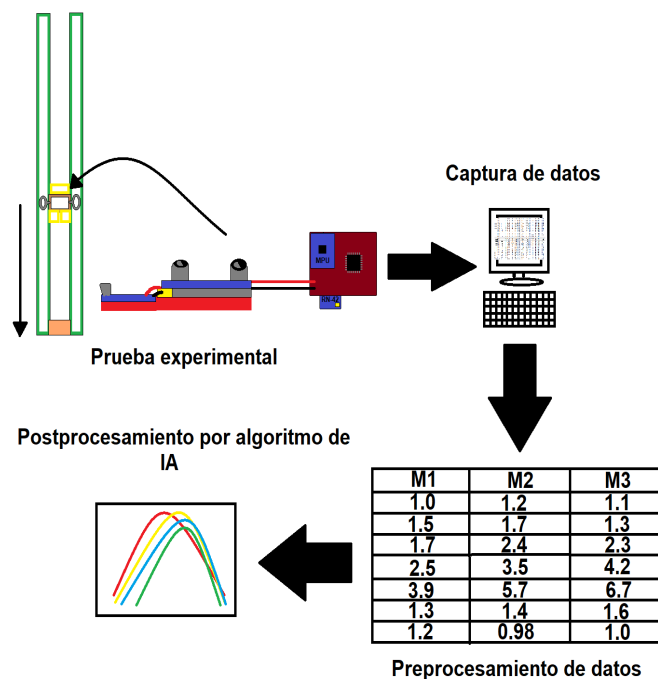


Figura 3.30: Esquema general de prueba experimental para configuración del sistema embebido de monitoreo inercial.

Para simular eventos de caída libre en condiciones controladas, se implementó una estructura experimental compuesta por una base móvil que soporta el sistema embebido y se encuentra

inicialmente en reposo a una altura conocida.

A continuación, en la [Figura 3.31](#) se muestra la base de sobre la que se colocaría el sistema embebido junto con su módulo de alimentación, y una base de esponja para amortiguar el impacto contra la superficie del suelo.

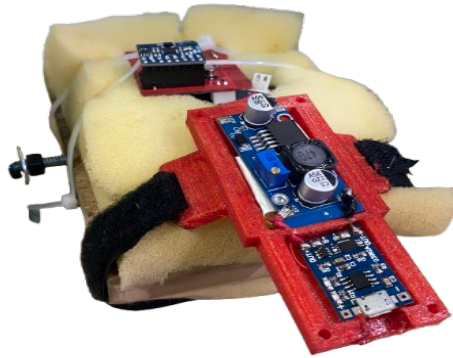


Figura 3.31: Base de soporte para sistema embebido de monitoreo inercial.

Dicha base se apoya de manera temporal sobre dos elementos de madera (colocados en los extremos delantero y trasero de la base), que actúan como soportes removibles con el objetivo de estandarizar la medición entre cada una de las pruebas a realizar, como se observa a continuación en la [Figura 3.32](#).



Figura 3.32: Esquema general de prueba experimental para configuración del sistema embebido de monitoreo inercial.

Como se puede observar en la figura anterior, la estructura consta de dos rieles para soportar la base del sistema embebido, la cual, en caída libre, se deslizará a través de los rieles de la estructura experimental.

Los soportes removibles de la base experimental en la que se encuentra el sistema embebido se colocaron a una altura de 0.4 metros a partir de la base inferior de la estructura experimental.

Antes de realizar las pruebas experimentales, fue necesario tomar mediciones respecto al peso de la base de soporte del sistema embebido, con el objetivo de caracterizar la prueba y comparar los resultados. Como se sabe, el peso del cuerpo es un factor importante al monitorear una caída libre mediante un MPU6050, ya que, aunque la aceleración gravitacional es constante e independiente de la masa, el peso influye en cómo el objeto interactúa con su entorno, especialmente cuando existe resistencia del aire o impactos posteriores a la caída. Un cuerpo más pesado tenderá a verse menos afectado por la fricción o turbulencia, generando mediciones más estables en el acelerómetro, mientras que uno más ligero puede presentar variaciones bruscas en los datos debido a desaceleraciones irregulares durante el descenso.

A continuación, en la [Tabla 3.4](#) se muestran los resultados del peso de la base experimental

del sistema embebido de monitoreo inercial.

Tabla 3.4: Medición del peso de la base de soporte para el sistema embebido de monitoreo inercial.

PESO (g)	M1	M2	M3	M4	M5	PROMEDIO
BASE	184.2	184.4	184.5	184.4	184.3	184.36
BASE + ARREGLO	562.5	563.3	563.4	562.3	563.5	563.00
ARREGLO	378.3	378.9	378.9	377.9	379.2	378.64

Como se puede analizar en la [Tabla 3.4](#), la medición consistió en pesar cada uno de los elementos que conforman el arreglo de la base con los soportes para el sistema embebido y sus piezas de protección.

Como primera instancia, se realizaron una serie de cinco mediciones. Los datos inerciales fueron registrados en un archivo CSV gracias a la herramienta PLX-DAQ de Excel, comunicándose por el puerto serial mediante un módulo RN42 con Bluetooth 2.0 en el sistema embebido de monitoreo inercial.

A continuación, en la [Figura 3.33](#) se presenta un esquema de los componentes necesarios para el proceso de registro de datos mediante la herramienta de Excel PLX-DAQ.

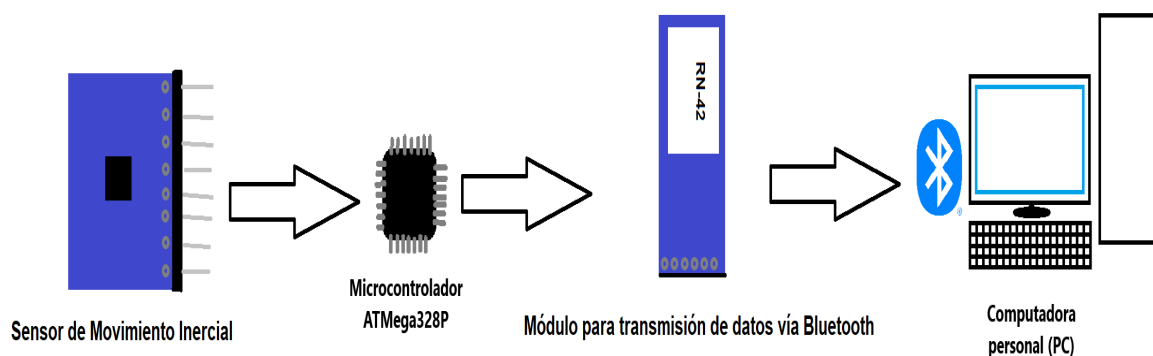


Figura 3.33: Esquema general de los componentes fundamentales para la transmisión de datos inerciales a la herramienta de registro.

Como se puede analizar en la figura anterior, el sensor inercial registra los datos a través del microcontrolador, y este, a su vez, los procesa y los transmite a la herramienta de registro de una computadora personal a través del puerto serial mediante un módulo de comunicación Bluetooth; en este caso, el RN42 con Bluetooth 2.0.

Como se mencionó anteriormente, el sensor fue programado por interrupciones con el objetivo de estandarizar cada una de las mediciones durante esta prueba. Por lo tanto al remover los soportes de madera de la estructura experimental, la base del sistema embebido caería debido a la fuerza de gravedad, deslizándose a través de los rieles mediante las guías metálicas de deslizamiento lineal atornilladas en la base de soporte del sistema embebido de monitoreo inercial.

En el primer ensayo, se registró la aceleración resultante de 5 mediciones del experimento. A continuación se presenta la interfaz de captura para una de las mediciones del proceso experimental de la herramienta en Excel. (Ver [Figura 3.34](#))

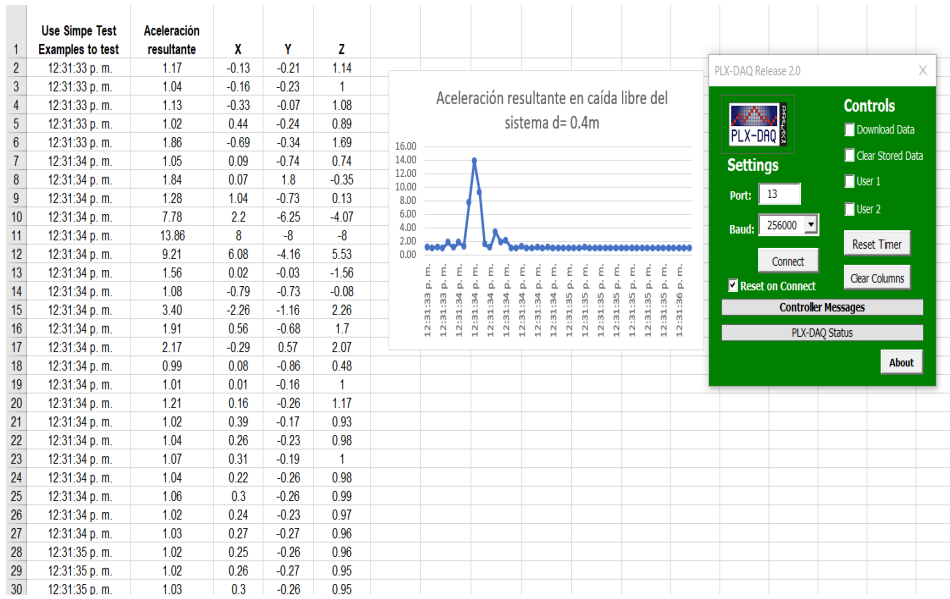


Figura 3.34: Interfaz de registro de datos inerciales de herramienta PLX-DAQ en Excel 2016.

El objetivo del experimento se basa en la validación del algoritmo de inteligencia artificial, en este caso programado en Python, para la identificación de patrones de datos de acelera-

ción resultante en eventos de movimiento inercial mediante Clustering por K-Means. Este algoritmo nos permite confirmar e identificar diferentes eventos de aceleración gracias a la clasificación Clustering. Para ello se registraron 20 mediciones, de las cuáles 10 se registraron sin amortiguación, es decir, se registró la aceleración resultante alcanzada en un impacto directo del sistema embebido en caída libre, la base inferior de la estructura experimental, mientras que las otras 10 se registraron con amortiguación .

Durante la calibración del sensor se optó por configurar el acelerómetro en una sensibilidad de 8g, ya que al estar configurado de fábrica en 2g, el sensor se saturaba al momento de impactar con la superficie, por lo que no era tan precisa la medición de la aceleración resultante ocasionada por el impacto.

A continuación, en la [Figura 3.35](#) se muestra el gráfico generado mediante la librería de Matplotlib para el Clustering por K-means.

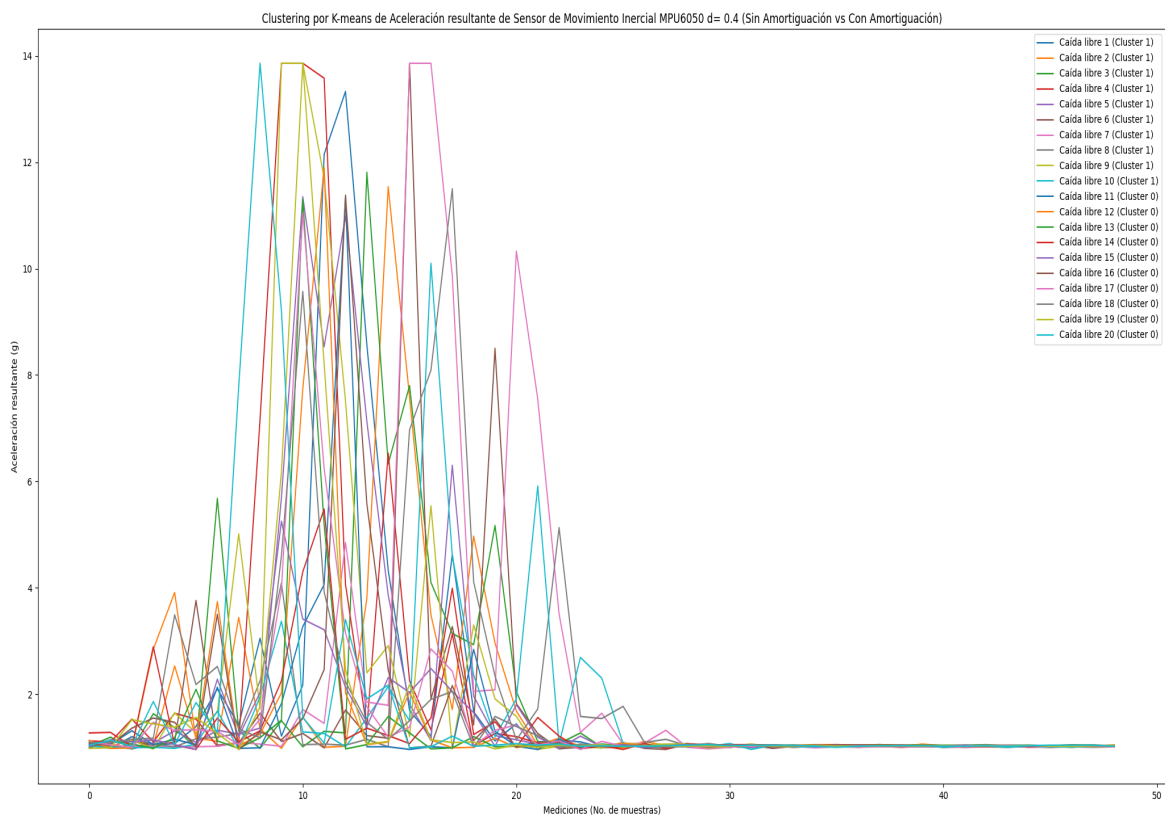


Figura 3.35: Gráfica 1 de Clustering por K-Means para 20 grupos de datos de aceleración resultante en caída libre.

Como se puede ver en la [Figura 3.35](#), el algoritmo de Clustering por K-Means fue capaz de clasificar los datos de ambos tipos de caídas en grupos diferentes, marcando una precisión satisfactoria.

En la Figura podemos ver que los primeros 10 grupos de datos correspondientes a las pruebas experimentales sin amortiguación fueron clasificados en el cluster 1, mientras que los 10 grupos de las pruebas con amortiguación fueron clasificados en el cluster 0.

Para esta ejecución del algoritmo, los datos fueron procesados en el orden tal cual fueron medidos: primero los 10 grupos de datos sin amortiguación y posteriormente los 10 grupos de datos con amortiguación.

Como paso siguiente para validar la eficacia del algoritmo en la clasificación, se propuso cambiar el orden de los grupos de datos en el Preprocesamiento para comprobar que el Clustering por K-means es capaz de clasificar los grupos de datos en distinto orden. (Ver [Figura 3.36](#))

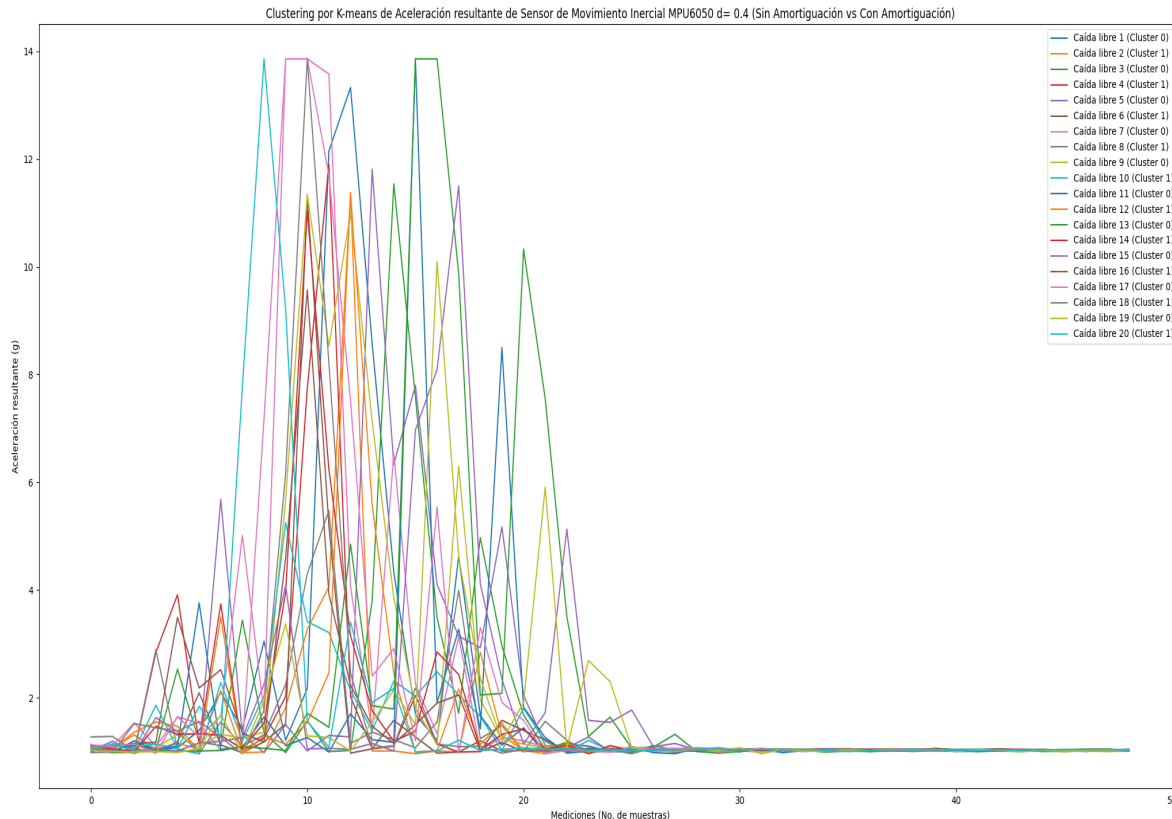


Figura 3.36: Gráfica 2 de Clustering por K-Means para 20 grupos de datos de aceleración resultante en caída libre.

En la [Figura 3.36](#) podemos ver que ahora, el algoritmo fue clasificando los grupos de datos en clusters alternados ya que al cambiar el orden del ingreso de los datos en el preprocesamiento, alteramos el orden en el que debería mostrar los resultados el algoritmo; sin embargo, se puede observar que el algoritmo mantiene su precisión para identificar en clusters diferentes los grupos de datos de las pruebas sin amortiguación, en este caso en el cluster 0, y los grupos de datos con amortiguación en el cluster 1.

Con este experimento validamos el funcionamiento del sistema embebido de monitoreo inercial para clasificación de datos de aceleración en caída libre mediante una estructura experimental.

Posteriormente, se realizaron pruebas de configuración para el giroscopio, ya que su configuración de fábrica es de ± 250 grados por segundo ($^{\circ}/s$), con una sensibilidad de 131 Least Significant Bits por grado por segundo (LSB/ $^{\circ}/s$), y una configuración de 2 g para el acele-

rómetro, es decir, una sensibilidad de 16.384 Least Significant Bits por unidad de gravedad (LSB/g). Eso representaba un rango muy bajo para movimientos en los que se involucran tanto aceleraciones como velocidades angulares con cambios drásticos en muy poco tiempo. Por ello, se seleccionó la configuración de ± 1000 °/s como rango de medición para el giroscopio, con una sensibilidad de 32.8 LSB/°/s, con el propósito de evitar saturaciones de los sensores en el registro de datos en movimientos repentinos o intensos.

Concluida la etapa 4, se logró implementar y validar satisfactoriamente el sistema embebido de monitoreo inercial, utilizando una estructura de caída libre controlada como entorno experimental para comprobar su capacidad de adquisición y transmisión precisa de datos de aceleración. Esta validación experimental no solo confirma el correcto funcionamiento del sistema, sino que también sienta las bases para avanzar hacia la etapa 5, en la cual se abordará el desarrollo de una Red Neuronal para el reconocimiento de actividades humanas (HAR). En esta nueva fase, se seleccionarán los movimientos específicos a registrar, se construirá la base de datos correspondiente y se diseñará el algoritmo que permitirá la creación y entrenamiento de la Red Neuronal, integrando así inteligencia artificial al sistema de monitoreo previamente desarrollado.

3.5 METODOLOGÍA PARA EL DESARROLLO DE UNA RED NEURONAL PARA INFERENCIA INERCIAL (Etapa 5)

En esta etapa se aborda el diseño y desarrollo de la red neuronal destinada al reconocimiento de actividades humanas, la cual constituye el núcleo del sistema neuromórfico propuesto. El proceso inicia con el análisis estratégico de la colocación de sensores inerciales, buscando una configuración óptima que permita una recolección precisa y representativa de los movimientos.

Posteriormente, se realiza la selección de las actividades cotidianas más relevantes a registrar, considerando criterios funcionales y de aplicabilidad en contextos de asistencia a adultos mayores. A partir de esto, se lleva a cabo la captura de una base de datos, seguida de un postprocesamiento para asegurar la calidad y la uniformidad de los datos. Finalmente, se desarrolla el algoritmo para formar la red neuronal, considerando su futura implementación eficiente dentro de la arquitectura del sistema neuromórfico.

3.5.1 Análisis teórico para ubicación del sistema embebido de monitoreo inercial

La ubicación del sensor inercial es un factor crítico para el diseño y desempeño de sistemas de reconocimiento de actividades humanas, especialmente en el contexto de herramientas de asistencia para adultos mayores, como se propone en la presente tesis. La correcta colocación del dispositivo determina la calidad y confiabilidad de los datos recolectados, lo que a su vez impacta directamente en la precisión de los modelos de clasificación basados en aprendizaje automático. Una posición mal seleccionada puede introducir ruido o variabilidad en las mediciones, dificultando la detección de patrones consistentes y reduciendo la capacidad del sistema para diferenciar actividades con características cinemáticas similares, como sentarse, levantarse o caminar.

A continuación, en la [Figura 3.37](#) se presenta la orientación de los ejes del acelerómetro y del giroscopio de la unidad de medición inercial empleada en el presente trabajo de investigación.

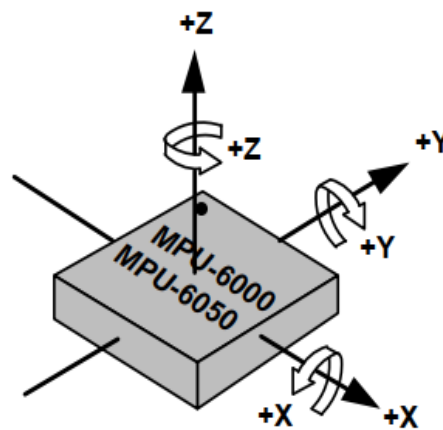


Figura 3.37: Representación y orientación de ejes de sensor MPU6050.

La colocación de la unidad de medición inercial (IMU) sobre el pectoral derecho resulta óptima para el reconocimiento de actividades humanas debido a factores relacionados con la seguridad del usuario, la orientación adecuada de los ejes de medición y la reducción de riesgos mecánicos y fisiológicos durante el uso. Desde el punto de vista biomecánico, el pecho es una región central y estable del torso, lo que permite registrar de manera más precisa los movimientos del cuerpo sin verse afectado por los movimientos simples de las extremidades,

como los brazos o las piernas, que podrían introducir ruido en las mediciones. Ubicar el sensor sobre el pectoral derecho, además, facilita la estandarización de las mediciones, asegurando que todos los registros se realicen con la misma referencia anatómica, lo que es esencial para entrenar modelos de reconocimiento de actividades con alta confiabilidad.

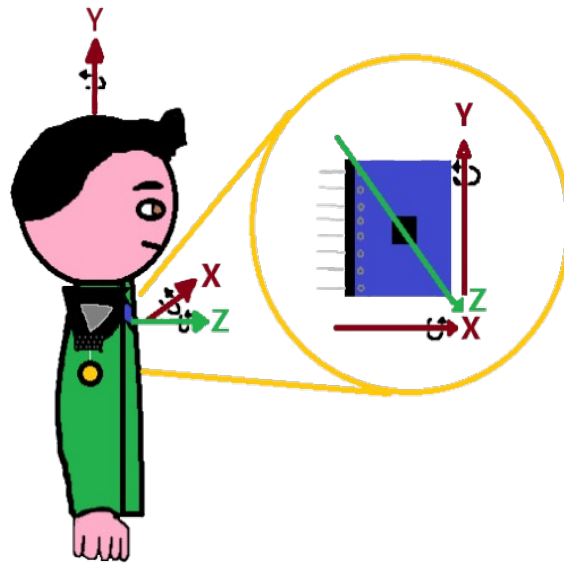


Figura 3.38: Representación de la colocación del sistema embebido de HAR como herramienta de asistencia.

En términos de seguridad, esta ubicación evita zonas de alto riesgo como la cabeza, la columna vertebral o las articulaciones, donde un impacto o caída del dispositivo podría ocasionar lesiones. Además, al situarse sobre el músculo pectoral y no directamente sobre el esternón, se reduce la presión sobre órganos vitales como el corazón y los pulmones en caso de impacto, al mismo tiempo que se brinda una superficie plana y ligeramente acolchada que favorece una fijación firme mediante un arnés o cinta elástica. Esto minimiza el riesgo de irritaciones en la piel y evita restricciones en la respiración, asegurando la comodidad del usuario durante períodos prolongados de monitoreo.

La orientación de los ejes del sensor también se beneficia de esta ubicación. Al colocarse en el pecho, se obtiene una alineación consistente con los ejes anatómicos del cuerpo: el eje X se alinea con el movimiento lateral (izquierda-derecha), el eje Y con el desplazamiento vertical (arriba-abajo) y el eje Z con la dirección anteroposterior (adelante-atrás). Esta configuración

simplifica la interpretación de los datos, permitiendo que las variaciones en aceleración y velocidad angular correspondan directamente a movimientos corporales naturales, lo cual es crítico para la clasificación de actividades como caminar, sentarse, levantarse o subir y bajar escaleras.

Finalmente, esta posición reduce los riesgos asociados a la caída del dispositivo o a la interferencia en actividades cotidianas. Al estar ubicada en una zona protegida y cercana al centro de masa del cuerpo, disminuye la probabilidad de enganches con objetos externos y la exposición a impactos directos en comparación con ubicaciones periféricas, como muñecas, tobillos o cintura. Además, el movimiento controlado de esta región del torso contribuye a prolongar la vida útil de los componentes electrónicos, ya que se reduce la vibración excesiva que podría afectar su desempeño.

En conjunto, la ubicación de la IMU sobre el pectoral derecho ofrece un balance ideal entre seguridad, calidad de datos y comodidad del usuario, factores determinantes para el desarrollo de un sistema confiable de reconocimiento de actividades humanas orientado a la asistencia y monitoreo de adultos mayores.

3.5.2 Selección de movimientos y captura de base de datos para entrenamiento

Las actividades físicas a analizar se proponen en función de las actividades más comunes en el día a día de una persona de la tercera edad, como lo son: caminar, levantarse de un asiento, sentarse en un asiento, subir un escalón, bajar un escalón e incluso eventos específicos, como puede llegar a ser una caída.

Para tener un buen desempeño en el análisis de las señales proporcionadas por el sistema embebido de monitoreo, a continuación se define en detalle la actividad física correspondiente a la clase que se desea analizar.

Es de suma importancia remarcar que las mediciones experimentales realizadas para este trabajo fueron registradas para un usuario con una estatura de 1.66 metros y un peso de 58 kg, siendo el peso y la altura factores clave en la medición de la aceleración y la velocidad angular.

CAMINAR: La medición consiste en monitorear la señal del acelerómetro y el giroscopio en sus tres ejes al dar un único paso, siendo el eje Z el predominante para esta clase.

LEVANTARSE: La presente categoría representa la acción física de levantarse de un asiento, siendo el eje Y y el eje Z los más representativos para la clase.

SENTARSE: La categoría representa la acción física de sentarse, siendo los ejes representativos el eje Y y el eje Z, ya que, como se puede observar en la Figura X, son los ejes en los cuales el usuario lleva a cabo la acción.

SUBIR ESCALÓN: La categoría representa la acción física de subir un escalón de 0.4 metros aproximadamente, primero colocando el pie de apoyo al subir el escalón, y posteriormente subiendo el otro pie para equilibrio. En dicha categoría, los ejes más representativos son el eje Y, y, a su vez, el eje Z por el cambio de proyección en la parte posterior del usuario.

BAJAR ESCALÓN: La categoría representa la acción física de bajar un escalón de aproximadamente 0.4 metros, primero colocando el pie de apoyo al descender del escalón y, posteriormente, bajando el otro pie para mantener el equilibrio. En dicha categoría, los ejes más representativos son el eje Y y, a su vez, el eje Z, por el cambio de proyección en la parte posterior del usuario.

AGACHARSE: La categoría representa la acción física de inclinar el tronco hacia adelante para reducir la altura del cuerpo sin llegar a sentarse, manteniendo los pies en contacto con el suelo. Durante este movimiento, los ejes más representativos son el eje Y y el eje Z, debido a los cambios en la proyección vertical y el desplazamiento hacia adelante del usuario. Estos ejes permiten identificar claramente la transición desde la posición erguida hasta la postura agachada.

CAÍDA: La categoría representa el evento físico de una caída lateral hacia cualquiera de los dos lados, siendo de suma importancia el correcto análisis de los tres ejes del acelerómetro y el giroscopio, ya que el movimiento representa un cambio significativo tanto en aceleración como en posición del sensor o, en este caso, el tronco del usuario.

Para el registro de la base de datos, se realiza el mismo procedimiento que se utilizó en

la metodología de la etapa 4, la cuál consistió en el desarrollo del sistema embebido de monitoreo inercial, en el que se configuró el sensor para registrar movimientos repentinos y evitar saturaciones en el acelerómetro o en el giroscopio.

Para esta etapa se seleccionaron un total de 11 clases de las 16 propuestas al inicio, ya que, al ser una herramienta enfocada a los adultos mayores, clases como subir escalones o bajar escalones son inusuales en este público, por lo que se descartaron dichas clases, a fin de mejorar la precisión del modelo para inferir entre las demás categorías. Como complemento para la base de datos, se registraron 50 mediciones del movimiento de Agacharse.

Del mismo modo, el sensor fue configurado para transmitir datos una vez recibida la interrupción por parte del microcontrolador al superar un umbral de aceleración, indicando que el movimiento ha comenzado. Se registraron 50 mediciones por cada movimiento seleccionado, siendo que cada medición estaría formada por 100 muestras de bloques de 8 datos, de modo que cada bloque tendría la siguiente estructura: Aceleración Resultante, Aceleración en Eje X, Aceleración en Eje Y, Aceleración en Eje Z, Velocidad Angular Resultante, Velocidad Angular en Eje X, Velocidad Angular en Eje Y, Velocidad Angular en Eje Z.

De modo que la estructura abreviada sería en bloques como: AR_n, Ax_n, Ay_n, Az_n, VR_n, Vx_n, Vy_n, Vz_n..., siendo n el número de muestras de la medición, considerando que se registrarían 100 muestras por cada medición del movimiento a registrar, por lo que se tendría un total de 800 datos por medición o movimiento a registrar. (Ver [Ecuación 3.1](#))

3.5.3 Postprocesamiento de los datos registrados

Una vez registrada la base de datos, con 50 mediciones de cada una de las 11 clases clasificadas, fue necesaria una metodología para dar formato a los datos y prepararlos para el entrenamiento de un modelo de Redes Neuronales.

La estructura de los datos en el registro fue propuesta a causa de una investigación sobre la mejor forma de procesar una cantidad de datos inerciales cronológicos considerable en la ejecución de cada uno de los movimientos, debido a que es de suma importancia conservar el orden cronológico del movimiento para evitar sesgos de información en la interpretación y/o

análisis de los resultados.

Durante la etapa de registro mediante la herramienta PLX-DAQ en Microsoft Excel, los datos registrados en bloques de 8 se presentaban de manera vertical, como se puede observar a continuación en la [Figura 3.39](#).

	Use Simpe Test Examples to test	Aceleración resultante	X	Y	Z	Velocidad angular resultante	X	Y	Z
1									
2	05:18:49 p. m.	1.06	0.74	-0.73	0.18	64.61	-37.9	-50.91	-12.07
3	05:18:49 p. m.	1.07	0.74	-0.75	0.16	62.91	-34.94	-50.73	-12.77
4	05:18:49 p. m.	1.24	0.91	-0.83	0.09	56.07	-35.79	-41.55	-11.65
5	05:18:49 p. m.	1.33	0.98	-0.9	-0.06	36.39	-19.36	-29.18	-9.91
6	05:18:49 p. m.	1.35	1	-0.89	-0.14	20.48	5.64	-19.63	1.43
7	05:18:50 p. m.	1.31	1	-0.83	-0.16	7.75	-4.18	-1.89	6.25
8	05:18:50 p. m.	1.2	0.85	-0.83	-0.13	20.87	11.34	7.77	15.7
9	05:18:50 p. m.	1.14	0.82	-0.8	-0.08	24.68	16.04	3.9	18.35
10	05:18:50 p. m.	1.13	0.83	-0.76	-0.02	30.46	20.82	15.37	16.07

Figura 3.39: Estructura de los bloques de datos registrados en la herramienta PLX-DAQ de Excel.

Con el propósito de realizar un análisis comparativo entre muestras de cada una de las mediciones de manera sencilla, se propuso, como primer paso del posprocesamiento, la reestructuración de los datos en forma horizontal.

Con el propósito de elaborar un modelo ligero enfocado en su implementación en microcontroladores, se desarrollaron algoritmos de reducción de datos con el objetivo de seleccionar las características o bloques de datos más importantes de cada una de las mediciones. Como primera propuesta, se elaboró un algoritmo para calcular la energía de los bloques de datos más representativos de cada medición mediante la sumatoria del valor absoluto.

Para disminuir la dimensionalidad de la base de datos inercial sin perder información significativa, se aplicó un método basado en la energía de bloques de señales. Cada muestra del conjunto original contiene 800 datos, que corresponden a mediciones consecutivas de aceleración y velocidad lineal registradas por los sensores inerciales.

1. Estructura de los bloques

Los 800 datos se agrupan en bloques de 8 valores, donde cada bloque B_n representa un instante

de tiempo n con las siguientes variables:

Ecuación 3.1: Estructura de los datos inerciales

$$B_n = \{AR_n, AX_n, AY_n, AZ_n, VR_n, VX_n, VY_n, VZ_n\} \quad (3.1)$$

donde:

- AR_n es la aceleración resultante,
- AX_n, AY_n, AZ_n son la aceleración en los ejes X, Y, Z,
- VR_n es la velocidad angular resultante,
- VX_n, VY_n, VZ_n son las velocidades angulares en los ejes X, Y, Z.

Como se mencionó antes, dado que cada bloque 8 datos y cada medición de la clase tendrá 800 datos debido a las 100 muestras, el

Ecuación 3.2: Fórmula para el número de "n" mediciones

$$n = \frac{800}{8} = 100 \quad (3.2)$$

Para el cálculo de energía, a cada bloque se le asocia un valor o "peso" que cuantifica la magnitud total del movimiento en ese intervalo. Este valor se obtiene mediante la sumatoria de los valores absolutos de cada una de las variables del bloque, de modo que:

Ecuación 3.3: Fórmula para el cálculo del peso o energía del bloque de datos

$$w_n = |AR_n| + |AX_n| + |AY_n| + |AZ_n| + |VR_n| + |VX_n| + |VY_n| + |VZ_n| \quad (3.3)$$

Este parámetro permite identificar las secciones de la señal donde ocurre un mayor cambio de aceleración y/o velocidad angular en la actividad física, ya que los cambios bruscos de estas variables producen un incremento directamente en w_n .

Una vez calculados los bloques más representativos correspondientes a cada uno de ellos, se seleccionaron, como primera prueba, los 50 bloques con mayor energía, con el propósito de reducir la cantidad de datos a procesar y en que el modelo sea lo suficientemente compacto como para ser implementado en un microcontrolador.

La selección se realiza ordenando los pesos de forma descendente y tomando los índices correspondientes a los veinte valores más altos, de modo que:

Ecuación 3.4: Orden para la selección de los bloques de datos más representativos.

$$B_{j1}, B_{j2}, \dots, B_{j50} \text{ con } w_{j1} \geq w_{j2} \geq \dots \geq w_{j50} \quad (3.4)$$

Posteriormente, estos bloques se reordenan en su secuencia temporal original para conservar la coherencia de la señal; cada uno de los 50 bloques seleccionados conserva sus 8 datos en la estructura de [Ecuación 3.1](#).

De esta forma es que se obtiene la nueva muestra, concatenando los bloques seleccionados en el orden temporal original del movimiento inercial:

Ecuación 3.5: Estructura de la nueva muestra de 50 bloques de datos más representativos.

$$X_r = [B_{j1} || B_{j2} || \dots || B_{j50}] \quad (3.5)$$

donde || representa la operación de concatenación. El tamaño total de la nueva muestra es:

Ecuación 3.6: Cálculo de datos totales por cada muestra o medición de la clase.

$$T = (50)(8) = 400 \quad (3.6)$$

De esta forma, cada muestra de 800 datos se transforma en una versión reducida de 400 datos, concentrando únicamente los intervalos de mayor actividad. Este procedimiento disminuye el tamaño del conjunto de datos en un 50 % sin eliminar la información esencial para el análisis de patrones de movimiento o para la etapa de inferencia en el sistema neuromórfico.

Con el propósito de evaluar distintos modelos, se utilizó otra técnica de selección de características más importantes, denominada XGBoost.

XGBoost, que se refiere a Extreme Gradient Boosting, es una técnica comúnmente utilizada en algoritmos que cuentan con una base de datos significativa.

Esta técnica entrena un conjunto de árboles de decisión que se van sumando secuencialmente para minimizar una función de pérdida.

Matemáticamente, si tenemos un conjunto de datos:

El proceso de selección de características mediante *XGBoost* se basa en la construcción iterativa de árboles de decisión, donde cada árbol busca corregir los errores cometidos por los anteriores. A continuación se describe matemáticamente este proceso y la forma en que se calcula la importancia de cada característica.

Ecuación 3.7: Conjunto de datos de entrenamiento

$$D = \{(x_i, y_i)\}_{i=1}^N, \quad x_i \in \mathbb{R}^m, y_i \in \mathcal{Y} \quad (3.7)$$

La ecuación (3.7) representa el conjunto de datos de entrenamiento, donde cada ejemplo x_i contiene m características y y_i es la etiqueta o salida asociada. Este conjunto constituye la base para el aprendizaje supervisado del modelo.

Ecuación 3.8: Ecuación para predicción por suma de árboles de decisión

$$\hat{y}_i = \sum_{k=1}^K f_k(x_i), \quad f_k \in \mathcal{F} \quad (3.8)$$

En la ecuación (3.8), la predicción final $\hat{y}_i$ se obtiene como la suma de las salidas de K árboles de regresión. Cada árbol f_k pertenece al conjunto de funciones $\mathcal{F}$ que representan árboles de decisión con pesos en sus hojas.

Ecuación 3.9: Ecuación de función objetivo para cada iteración t

$$\text{Obj}^{(t)} = \sum_{i=1}^N l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \quad (3.9)$$

La ecuación (3.9) define la función objetivo que se minimiza en cada iteración t . Esta función incluye un término de pérdida l que mide el error entre la predicción actual y el valor real, así como un término de regularización $\Omega(f_t)$ que penaliza la complejidad del árbol para evitar sobreajuste.

Ecuación 3.10: Término de regularización del árbol

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2 \quad (3.10)$$

El término de regularización (3.10) considera tanto el número de hojas T del árbol como los pesos w_j asociados a cada una de ellas. Los hiperparámetros γ y λ controlan la penalización del tamaño y los valores de los pesos respectivamente.

Ecuación 3.11: Aproximación de segundo orden (Taylor)

$$\text{Obj}^{(t)} \approx \sum_{i=1}^N \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2 \right] + \Omega(f_t) \quad (3.11)$$

Para optimizar el modelo de manera eficiente, XGBoost utiliza una expansión de segundo orden (serie de Taylor) como se muestra en (3.11). En esta expresión, g_i y h_i representan el gradiente y la hessiana (segunda derivada) de la función de pérdida, lo que permite una optimización más precisa y estable.

Ecuación 3.12: Fórmulas del gradiente y la segunda derivada (hessiana) de la función de pérdida

$$g_i = \frac{\partial l(y_i, \hat{y}_i)}{\partial \hat{y}_i} \Big|_{\hat{y}_i = \hat{y}_i^{(t-1)}}, \quad h_i = \frac{\partial^2 l(y_i, \hat{y}_i)}{\partial \hat{y}_i^2} \Big|_{\hat{y}_i = \hat{y}_i^{(t-1)}} \quad (3.12)$$

La ecuación (3.12) define formalmente el gradiente y la hessiana locales para cada muestra i . Estos valores determinan la dirección y magnitud de los ajustes que el nuevo árbol debe realizar sobre las predicciones previas.

Ecuación 3.13: Ecuación para el cálculo de ganancia en nodo de árbol

$$\text{Gain}_s = \frac{1}{2} \left(\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right) - \gamma \quad (3.13)$$

La ecuación (3.13) mide la ganancia (*gain*) obtenida al realizar una división en un nodo del árbol. Esta ganancia indica cuánto mejora la función objetivo al dividir un conjunto de datos en dos ramas, izquierda (L) y derecha (R).

Ecuación 3.14: Ecuaciones para el cálculo de gradientes y hessianas de cada lado de partición

$$G_L = \sum_{i \in I_L} g_i, \quad H_L = \sum_{i \in I_L} h_i, \quad G_R = \sum_{i \in I_R} g_i, \quad H_R = \sum_{i \in I_R} h_i \quad (3.14)$$

Las expresiones de (3.14) muestran cómo se calculan las sumas acumuladas de gradientes y hessianos en cada lado de la partición. Estos valores se utilizan en el cálculo de la ganancia para determinar si la división aporta valor al modelo.

Ecuación 3.15: Expresión para el cálculo de la importancia total de una característica j

$$I_j = \sum_{t=1}^K \sum_{s \in \mathcal{S}_{t,j}} \text{Gain}_{t,s} \quad (3.15)$$

En (3.15), se define la importancia total de una característica j como la suma de todas las ganancias obtenidas por las divisiones que utilizan dicha característica en todos los árboles del modelo. Cuanto mayor sea esta suma, mayor es la relevancia de la característica en la predicción.

Ecuación 3.16: Expresión para el cálculo de la importancia promedio de cada característica considerando distintos modelos

$$I_j^{\text{prom}} = \frac{1}{M} \sum_{m=1}^M I_j^{(m)} \quad (3.16)$$

Cuando se entrena un modelo con múltiples salidas (*multioutput*), la ecuación (3.16) permite calcular el promedio de la importancia de cada característica considerando los distintos modelos entrenados.

Ecuación 3.17: Fórmula para el ordenamiento de las características según su importancia

$$\pi = \text{argsort}(I^{\text{prom}}) \quad (\text{orden descendente}) \quad (3.17)$$

Ecuación 3.18: Fórmula para la selección de las características N más relevantes

$$S_N = \{\pi(1), \pi(2), \dots, \pi(N)\} \quad (\text{índices de las } N \text{ mejores características}) \quad (3.18)$$

Las ecuaciones (3.17) y (3.18) muestran el proceso de ordenamiento de las características según su importancia y la selección de las N más relevantes para reducir la dimensionalidad

del conjunto de datos.

Ecuación 3.19: Representación de la construcción de la matriz de datos seleccionadas por XGBoost

$$X_{\text{top-}N} = X_{:,S_N} \quad (3.19)$$

Las ecuaciones presentadas durante el desarrollo de la sección del capítulo fueron tomadas de [32]

Finalmente, la ecuación (3.19) representa la construcción de la nueva matriz de datos con las N características más importantes seleccionadas por XGBoost. Esta matriz reducida se utiliza posteriormente como entrada al modelo de red neuronal o a cualquier otro clasificador, mejorando el rendimiento y reduciendo la complejidad computacional.

3.5.4 Derarrollo del algoritmo para la creación y entrenamiento del modelo de Red Neuronal

Para la creación del modelo de Red Neuronal, se utilizó un entorno virtual creado en Anaconda3, con el objetivo de facilitar la instalación de librerías y complementos específicos de Python necesarios.

El modelo de clasificación basado en redes neuronales fue desarrollado utilizando el lenguaje de programación Python y las bibliotecas TensorFlow y Keras. Dichas herramientas permitieron construir y entrenar un modelo capaz de reconocer distintos tipos de movimiento a partir de las características extraídas de la base de datos denominada BaseDatos_160.csv.

Inicialmente, se cargó la base de datos completa y se separaron las características de entrada (variables inerciales y derivadas) de la etiqueta de salida correspondiente a la clase de movimiento (MOV_OUT). Las etiquetas categóricas fueron codificadas numéricamente mediante un codificador de etiquetas (LabelEncoder) y, posteriormente, transformadas a formato one-hot encoding, lo cual permitió representar cada clase como un vector binario compatible con la salida de la red neuronal.

Los datos fueron divididos en dos subconjuntos: 80 % para entrenamiento y 20 % para prueba, utilizando una semilla aleatoria fija (random state = 42) para garantizar la reproducibilidad de los resultados.

Posteriormente, las características fueron normalizadas utilizando el método StandardScaler, de modo que cada variable presentara media cero y desviación estándar unitaria. Esta etapa fue esencial para evitar que variables con escalas distintas influyeran de forma desigual durante el entrenamiento.

El modelo se implementó bajo un esquema secuencial compuesto por capas densamente conectadas. La capa de entrada recibe 158 características, correspondientes al número total de variables seleccionadas tras el preprocesamiento.

La arquitectura incluye tres capas ocultas con funciones de activación ReLU, intercaladas con capas de regularización tipo Dropout (con una probabilidad del 10 %) para reducir el riesgo de sobreajuste. Asimismo, se aplicó una penalización L2 ($\lambda = 0.001$) a los pesos de ciertas capas para mejorar la generalización del modelo.

La capa de salida utiliza la función de activación Softmax, lo que permite asignar una probabilidad a cada una de las 11 clases de movimiento identificadas en la base de datos.

El entrenamiento se realizó utilizando el optimizador Adam, con una tasa de aprendizaje de 0.001 y la función de pérdida categorical cross-entropy. El modelo se entrenó durante 100 épocas con un tamaño de lote (batch size) de 32 muestras y un 10 % del conjunto de entrenamiento reservado para validación.

Se monitoreó la evolución de la función de pérdida y la precisión tanto en el conjunto de entrenamiento como en el de validación, lo que permitió verificar la correcta convergencia del modelo.

Una vez finalizado el entrenamiento, se evaluó el desempeño del modelo utilizando el conjunto de prueba. Entre las métricas empleadas se encuentran:

Precisión global (Accuracy): proporción de clasificaciones correctas.

Matriz de confusión: permitió identificar el comportamiento del modelo en cada categoría.

Reporte de clasificación: incluyó precisión (precision), exhaustividad (recall) y puntaje F1 (F1-score) para cada clase.

F1-score macro y ponderado, los cuales reflejaron el equilibrio del modelo entre las distintas clases.

Finalmente, el modelo entrenado fue guardado en formato .h5 bajo el nombre `movement_classifier.h5`, junto con el objeto de normalización (`scaler.pkl`), lo que permite su reutilización posterior en la etapa de inferencia en el sistema embebido.

Con el objetivo de comparar ambos modelos, se desarrolló de igual manera, un algoritmo para la creación de una Red neuronal, pero con una previa selección de las cincuenta características más relevantes de las once clases mediante XGBoost.

Este procedimiento permitió identificar las variables con mayor relevancia en la clasificación de actividades, garantizando que el modelo final trabajara únicamente con las características más representativas y evitando redundancias o ruido en los datos.

Para la estimación de la importancia de las características, se utilizó el modelo `XGBClassifier`, una implementación eficiente del algoritmo de gradient boosting optimizado para tareas de clasificación multiclase. Dado que el problema involucraba múltiples etiquetas de salida, se adoptó un esquema de clasificación multisalida (`MultiOutputClassifier`), lo cual permitió que un clasificador independiente de XGBoost se entrenara para cada clase del vector de salida.

Cada modelo individual generó un vector de importancias de características, y posteriormente se calculó el promedio de importancia entre todas las salidas, obteniendo un valor global representativo para cada variable de entrada.

De este modo, fue posible construir un ranking de características ordenadas de acuerdo con su grado de contribución al proceso de clasificación.

Tras el análisis, se determinaron las 50 características más influyentes en la clasificación de las actividades humanas. Estas variables fueron extraídas y ordenadas cronológicamente de acuerdo con el número identificador de cada característica, garantizando la coherencia temporal del conjunto de datos.

El resultado fue una nueva base de datos denominada BaseDatos_Top50.csv, que contiene exclusivamente las 50 características seleccionadas junto con la etiqueta de salida correspondiente. Este conjunto depurado fue posteriormente empleado para la creación y entrenamiento del modelo de red neuronal final mediante el algoritmo anterior.

Adicionalmente, el algoritmo generó un gráfico de barras con las 50 variables más importantes, mostrando visualmente la contribución relativa de cada una a la clasificación. Las características de mayor peso correspondieron principalmente a aquellas relacionadas con las aceleraciones y velocidades angulares, las cuales demostraron una fuerte correlación con las distintas actividades humanas.

En la presente sección se expuso de manera detallada la metodología empleada para el desarrollo del modelo de red neuronal, abarcando las etapas de preparación de datos, selección de características, definición de la arquitectura del modelo y configuración de los parámetros de entrenamiento. Asimismo, se describieron los procedimientos de evaluación aplicados para determinar el desempeño de los modelos desarrollados.

Los resultados derivados de las métricas de evaluación, las tablas de parámetros utilizados, los gráficos de precisión y pérdida, el análisis de correlación entre variables, las matrices de confusión y la gráfica correspondiente a las 50 características más relevantes se presentan y analizan de manera integral en el Capítulo 4: Resultados y conclusiones.

3.6 METODOLOGÍA PARA LA INTEGRACIÓN E IMPLEMENTACIÓN DE UNIDADES DEL SISTEMA NEUROMÓRFICO (Etapa 6)

En esta etapa se presentan los procedimientos y estrategias empleados para llevar a cabo la integración completa del sistema neuromórfico desarrollado, con el propósito de obtener un producto funcional que incorpore todos los elementos de hardware y software previamente diseñados. Esta fase representa la culminación del proceso metodológico, donde se transita desde el desarrollo experimental y la validación individual de componentes hacia la implementación integral en un sistema embebido operativo.

La Etapa 6 se estructura en cuatro subetapas que permiten abordar de forma ordenada el proceso de integración:

Implementación del modelo de red neuronal en un microcontrolador, en la cual se adapta y optimiza el modelo entrenado para su ejecución en un entorno de cómputo embebido con recursos limitados.

Diseño y manufactura del sistema embebido final, que comprende la elaboración del circuito impreso, la selección de componentes y el ensamblaje del hardware definitivo.

Integración del sistema neuromórfico, donde se vinculan los módulos sensoriales, de procesamiento y de comunicación para lograr la operación conjunta del sistema.

Validación funcional y pruebas finales, en la que se evalúa el desempeño global del dispositivo, verificando la precisión del modelo, la estabilidad del sistema y el cumplimiento de los objetivos planteados.

Con esta etapa se busca consolidar el desarrollo teórico y experimental en un prototipo completamente funcional, capaz de ejecutar el reconocimiento de actividades humanas en tiempo real mediante un enfoque neuromórfico implementado en hardware, sentando las bases para su futura optimización y escalabilidad hacia aplicaciones prácticas.

3.6.1 Implementación del modelo de Red Neuronal en microcontrolador

Conversión del modelo entrenado a formato compatible con el microcontrolador

Una vez obtenido y validado el modelo de red neuronal en formato .h5 (formato nativo de Keras), fue necesario convertirlo a un formato que pudiera ser interpretado por el microcontrolador, ya que estos dispositivos no cuentan con la capacidad de ejecutar modelos en dicho formato de manera directa. Para ello, se implementó un procedimiento de conversión que transforma el modelo a un archivo TensorFlow Lite (.tflite) y posteriormente a un archivo de encabezado C (.h) que contiene el modelo representado como un arreglo de bytes, listo para ser embebido en el código fuente del sistema mediante la biblioteca ArduTFLite.

El algoritmo utilizado para esta conversión se llevó a cabo en el entorno de Python, empleando

las librerías TensorFlow y Keras, siguiendo los pasos que se describen a continuación:

Se cargó el modelo previamente entrenado y almacenado en formato .h5 mediante la función `load_model()` de Keras, desactivando la compilación automática con el argumento `compile=False` para reducir el tiempo de carga y simplificar la conversión.

Utilizando la clase `TFLiteConverter` de TensorFlow, el modelo se transformó a un archivo .tflite, el cual está optimizado para su ejecución en dispositivos de recursos limitados.

Dado que el entorno de desarrollo de Arduino no puede leer directamente archivos binarios, el archivo .tflite se transformó en un arreglo de bytes en formato hexadecimal. Para ello, se leyó el contenido binario y se generó un archivo .h que define una constante de tipo `unsigned char`, conteniendo todo el modelo como un arreglo embebido en el código fuente.

El archivo generado (`movimientos_model.h`) contiene la estructura necesaria para ser incluida dentro del código del microcontrolador, donde el modelo puede ser cargado en memoria y ejecutado mediante la biblioteca `ArduTFLite`. Esta conversión permite que el sistema embebido ejecute inferencias de la red neuronal en tiempo real sin requerir conexión a un entorno de alto procesamiento, garantizando así la portabilidad y autonomía del sistema neuromórfico desarrollado.

Desarrollo e implementación del código para el sistema embebido del sistema neuromórfico para la inferencia

En esta etapa, se integró el modelo de red neuronal previamente entrenado y optimizado al entorno embebido, utilizando la librería `ArduTFLite`, la cual permite ejecutar modelos de TensorFlow Lite Micro (TFLM) sobre plataformas de hardware con recursos limitados, como el Arduino Portenta H7. El modelo fue convertido a un archivo binario denominado `model.h`, que contiene los pesos y la estructura de la red en formato C para su carga directa en la memoria del microcontrolador.

El bloque principal de inicialización del modelo se realiza en la función `setup()`, donde se define un área de memoria estática (`tensor arena`) de 3500 bytes, utilizada por el intérprete de TensorFlow Lite para la gestión de tensores y operaciones internas durante la inferencia:

A continuación, el modelo se carga mediante la función `modelInit(model_tflite, tensor_arena, kTensorArenaSize)`, que asocia la estructura binaria del modelo con el espacio de trabajo definido. Esta inicialización permite, posteriormente, ejecutar inferencias sin requerir asignación dinámica de memoria, asegurando estabilidad en la ejecución en tiempo real.

El sistema integra un sensor inercial MPU6050, encargado de adquirir datos de aceleración y velocidad angular en tres ejes. Este dispositivo se comunica mediante el protocolo I²C y genera una interrupción digital en el pin 10 del Portenta, configurada en modo rising edge para detectar eventos de movimiento:

Cada vez que se activa la interrupción, el sistema comienza una ventana temporal de adquisición de 2.5 segundos, en la que se registran 100 muestras compuestas por ocho características: aceleración resultante, componentes de aceleración en los ejes X, Y y Z, velocidad angular resultante y componentes del giroscopio.

Estas 100 muestras conforman un vector de 800 datos, almacenado en el arreglo `raw_data[800]`.

Para reducir el volumen de información y preservar los eventos más representativos del movimiento, se implementa la [Ecuación 3.3](#) para implementar procedimiento de selección basado en la energía de los bloques de señal. El sistema divide los 800 datos en 100 bloques de 8 elementos y calcula la energía de cada bloque como la suma del valor absoluto de sus componentes.

Los 20 bloques con mayor energía son seleccionados mediante la función `topKIndices()`, y sus datos se concatenan para formar un vector de 160 valores. Posteriormente, se eliminan el primer y último elemento, obteniendo un total de 158 características, las cuales corresponden a las entradas esperadas por el modelo neuronal.

Antes de la inferencia, los datos son normalizados aplicando las medias (`means[]`) y desviaciones estándar (`scales[]`) obtenidas mediante el algoritmo de la creación del modelo de red neuronal en Python.

Este proceso garantiza que las entradas se encuentren dentro del mismo rango estadístico que el modelo original, preservando la coherencia de las predicciones.

Una vez normalizados los datos, cada característica se asigna a la entrada del modelo mediante `modelSetInput(inputs[i], i)`. Posteriormente, la inferencia se ejecuta con `modelRunInference()`, la cual procesa los 158 valores a través de las capas internas de la red y genera once salidas correspondientes a las clases de actividades humanas que el sistema es capaz de reconocer, las cuales tienen un índice previamente asignado la creación del modelo:

Tabla 3.5: Índice de clases del modelo para el sistema neuromórfico

Índice	Clase
0	Agacharse
1	Caída lateral derecha
2	Caminar lento
3	Caminar normal)
4	Caminar rápido
5	Levantarse lento
6	Levantarse normal
7	Levantarse rápido
8	Sentarse lento
9	Sentarse normal
1	Sentarse rápido

El sistema identifica la clase de mayor probabilidad seleccionando el índice con el valor máximo de salida, es decir:

Ecuación 3.20: Fórmula para la inferencia de clase en microcontrolador.

$$\text{Clase predicha} = \arg \max_i (y_i) \quad (3.20)$$

La predicción se imprime por puerto serial y, en caso de detectar una caída (clase 1), el sistema ejecuta acciones de alerta automática, enviando un mensaje SMS y realizando una llamada al número predefinido mediante un módulo SIM800L conectado al Portenta H7.

Estas rutinas, `Enviar_msj()` y `realizarLlamada()`, utilizan comandos AT para gestionar la comunicación GSM, de modo que el sistema pueda alertar en tiempo real a un contacto de emergencia ante una detección de caída.

Con esta implementación, el microcontrolador ejecuta de manera autónoma todas las etapas del sistema neuromórfico:

Adquisición inercial.

Preprocesamiento de señales (energía y normalización).

Inferencia del modelo neuronal embebido.

Acción de respuesta ante eventos críticos.

De este modo, se logra una integración completa del modelo de aprendizaje automático en un entorno embebido de recursos limitados, sin requerir procesamiento externo, lo que permite su funcionamiento en tiempo real y de forma portátil.

3.6.2 Diseño y manufactura del sistema embebido final

En esta subetapa se aborda la construcción física y electrónica del sistema embebido que permitirá implementar de manera eficiente el modelo neuromórfico y los sensores asociados. El objetivo principal es traducir el diseño conceptual y funcional del sistema en un prototipo tangible, garantizando que cumpla con los requerimientos de rendimiento, confiabilidad y compactación establecidos en las etapas anteriores.

El desarrollo del sistema embebido final se estructura en tres fases principales. La primera consiste en el diseño del diagrama esquemático utilizando Multisim 14, donde se definen las conexiones entre los distintos componentes electrónicos, se establecen las fuentes de alimentación, y se verifican las interacciones entre los módulos de control y los sensores. Esta fase permite validar de manera virtual el comportamiento del circuito antes de su implementación física.

La segunda fase involucra el diseño de la placa de circuito impreso (PCB) en Ultiboard, donde se optimiza la distribución de los componentes y las rutas de conexión para minimizar

interferencias, pérdidas de señal y consumo energético, garantizando una estructura compacta y eficiente.

Finalmente, la tercera fase corresponde al proceso de manufactura, que comprende la fabricación de la PCB.

Esta etapa asegura que el prototipo sea funcional, robusto y listo para su integración con el modelo neuromórfico y los demás elementos del sistema.

En conjunto, estas fases permiten materializar el diseño conceptual del sistema, asegurando la fiabilidad y el rendimiento necesarios para las etapas de integración y validación funcional posteriores.

Diseño de la tarjeta de circuito impreso (PCB)

Como primera instancia se realizó el diagrama esquemático de conexiones del sistema embebido final, tomando en cuenta los diagramas esquemáticos de cada uno de los resultados preliminares de las etapas anteriores, adaptando ahora dichos diseños al microcontrolador utilizado para el sistema neuromórfico.

De la misma manera, el diagrama esquemático fue elaborado en el software de National Instruments Multisim 14.0 en su versión Power Pro Edition. A continuación, en la [Figura 3.40](#)

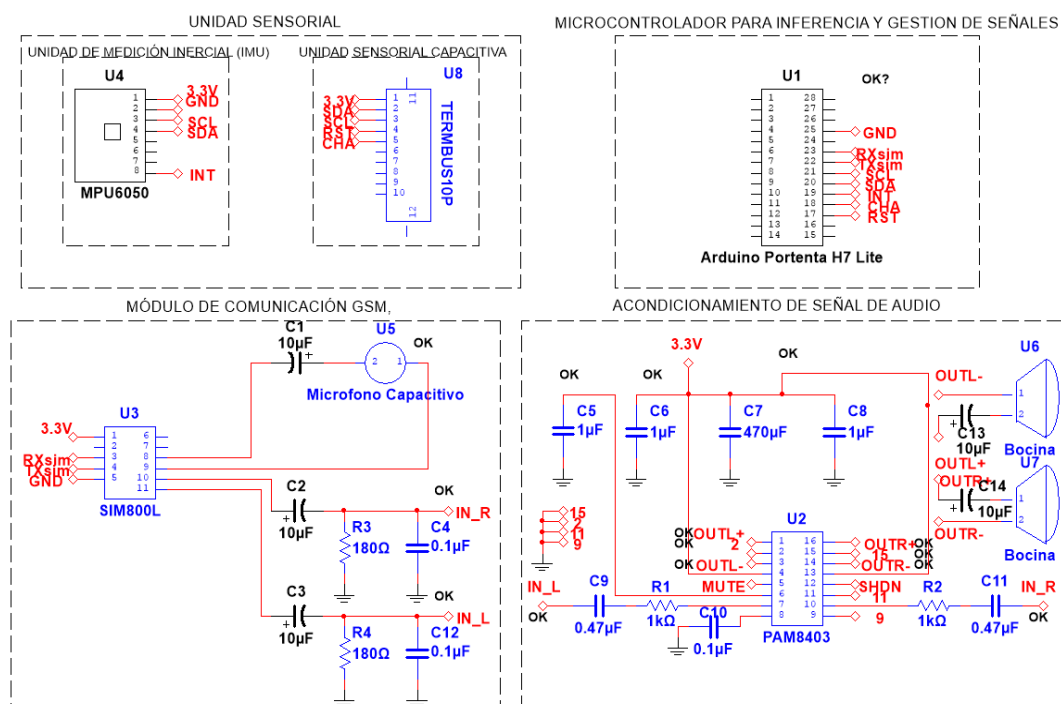


Figura 3.40: Diagrama esquemático del sistema embebido final para reconocimiento de actividades humanas en adultos mayores.

Como se puede observar en la figura anterior, el diagrama está integrado por 4 unidades, siendo la unidad sensorial dividida en dos partes, la unidad de medición inercial y la unidad sensorial capacitiva. Por otra parte, tenemos el módulo de comunicación GSM, encargado de establecer la comunicación en eventos críticos como las caídas. Dicho módulo posee su propio acondicionamiento de señal de audio en el cual, como se mencionó en la etapa 2 del desarrollo del proyecto, se utilizó un amplificador PAM8403.

Finalmente se integra el microcontrolador para inferencia y gestión de señales, el cual a su vez se conecta a las diferentes unidades sensoriales y de comunicación.

Una vez validado el diagrama esquemático de las tarjetas de circuito impreso se procedió a realizar el diseño de la tarjeta de circuito impreso en Ultiboard 14.0 respetando las normas aplicadas anteriormente.

A continuación en la [Figura 3.41](#) se presenta el diseño de la tarjeta de circuito impreso final para el sistema neuromórfico de reconocimiento de actividades humanas (HAR) en adultos

mayores.

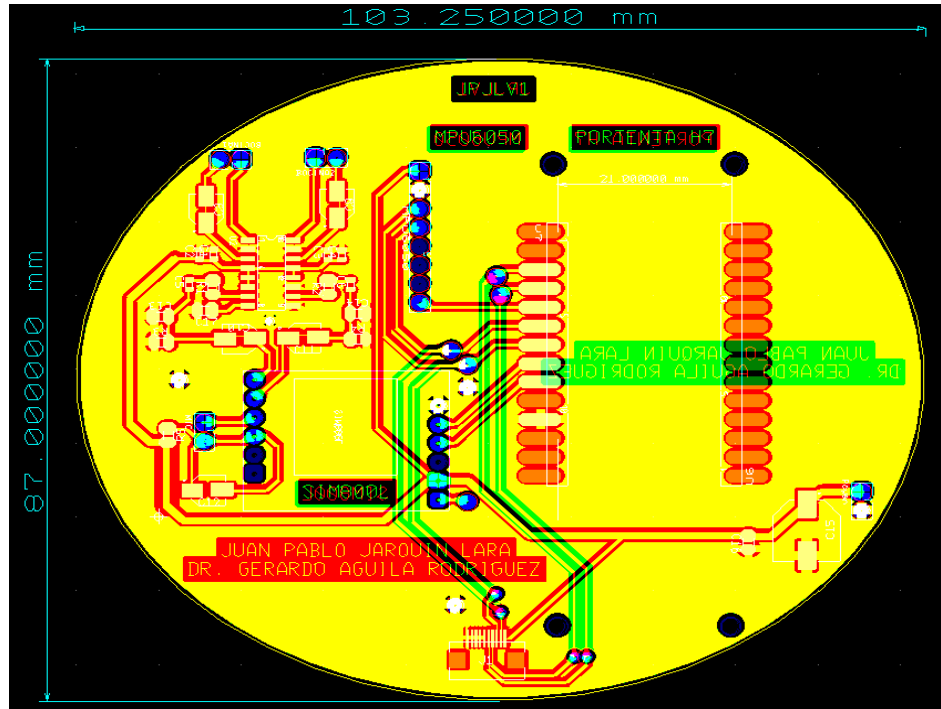


Figura 3.41: Diseño de tarjeta de circuito impreso del sistema embebido final para reconocimiento de actividades humanas en adultos mayores.

Con el propósito de ubicar de manera segura el sistema embebido, fue necesario proponer una geometría que se adaptara a la zona donde ubicaríamos el sistema embebido con la unidad de medición inercial. Es por ello que para el diseño de la tarjeta de circuito impreso final, se propone una geometría elíptica capaz de adaptarse con facilidad a la zona pectoral superior del usuario.

El diseño cuenta con dimensiones de aproximadamente 103.2 mm de largo por 87 mm de ancho. A pesar de ser una tarjeta de circuito impreso de un tamaño mediano a grande, se justifica debido al uso de la tarjeta de desarrollo Portenta H7, que ocupa aproximadamente más de un 30 % del tamaño de la tarjeta de circuito impreso.

Como se puede analizar en la [Figura 3.41](#), en el diseño del sistema embebido final se agregan en conjunto cada uno de los resultados preliminares de las etapas anteriores, es decir, la unidad sensorial, conformada por la unidad de medición inercial y los sensores capacitivos como parte

complementaria, la unidad de gestión de comunicación GSM; la unidad de acondicionamiento de audio; y la unidad de procesamiento de señales y de inferencia.

En el capítulo 4 de este trabajo, correspondiente a la presentación y discusión de los resultados, se comparten figuras y análisis de los resultados obtenidos en la manufactura del sistema embebido final.

3.6.3 Integración del sistema neuromórfico en Wearable

Para la integración del sistema embebido correspondiente al sistema neuromórfico con cada una de las unidades mostradas en la [Figura 2.1](#), fue necesario diseñar una pieza de soporte y protección para la tarjeta de circuito impreso. Con ese objetivo, se realizó un diseño en el software Fusion 360 de Autodesk. El diseño se realizó en base a la geometría de la tarjeta de circuito impreso mostrada en [Figura 3.41](#)

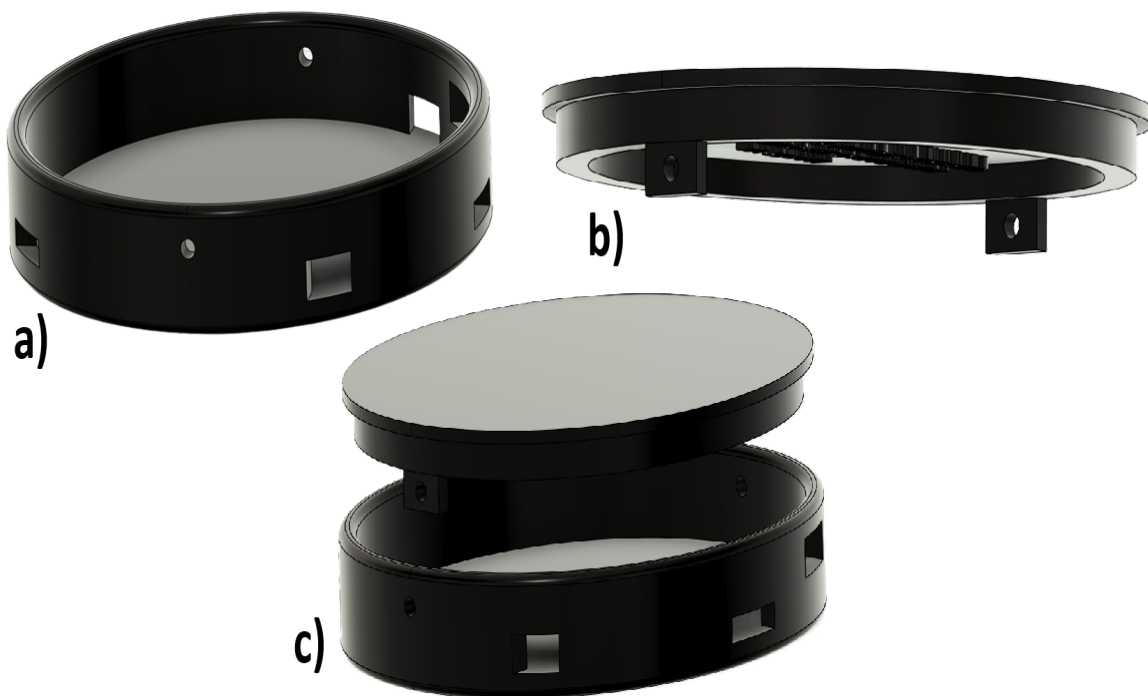


Figura 3.42: a) Diseño de base del enclosure para el sistema embebido. b) Diseño de tapa del enclosure para el sistema embebido. c) Vista isométrica del enclosure del sistema embebido.

La geometría del enclosure presentado en la [Figura 3.42](#) se diseñó tomando en cuenta restricciones de ergonomía y portabilidad, tomando en cuenta la seguridad tanto del sistema como del usuario.

Basandonos en esa metodología se elaboró de igual manera el enclosure para la unidad de alimentación del sistema, conformado por una batería recargable Li-Po de 3.7V a 2500 mAh con un módulo TP4056 para la recarga de la batería.

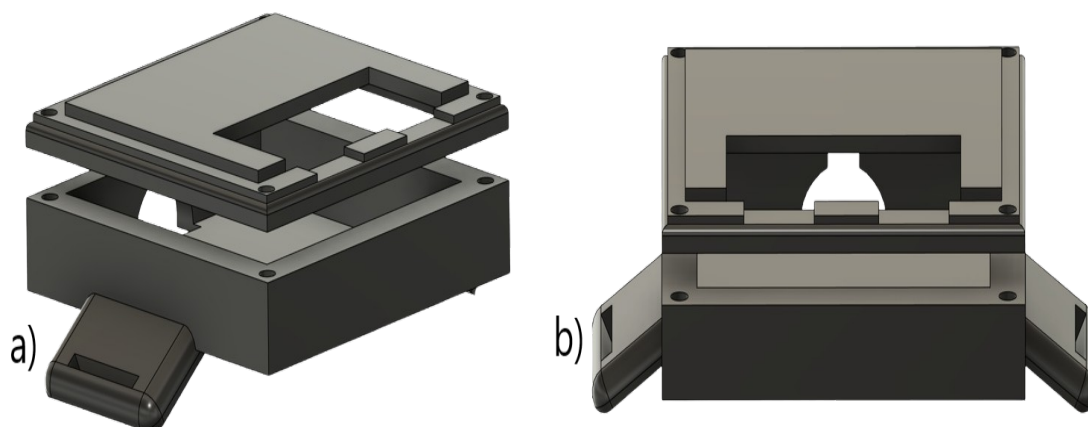


Figura 3.43: Diseño de base del enclosure para el sistema embebido

El enclosure propuesto en [Figura 3.43](#) se diseñó con el propósito de hacer de la unidad de alimentación adaptable a la morfología del usuario y al wearable propuesto para la herramienta.

A continuación, se muestra en la [Figura 3.44](#) la integración del sistema neuromórfico en la base del enclosure junto con su unidad de alimentación.

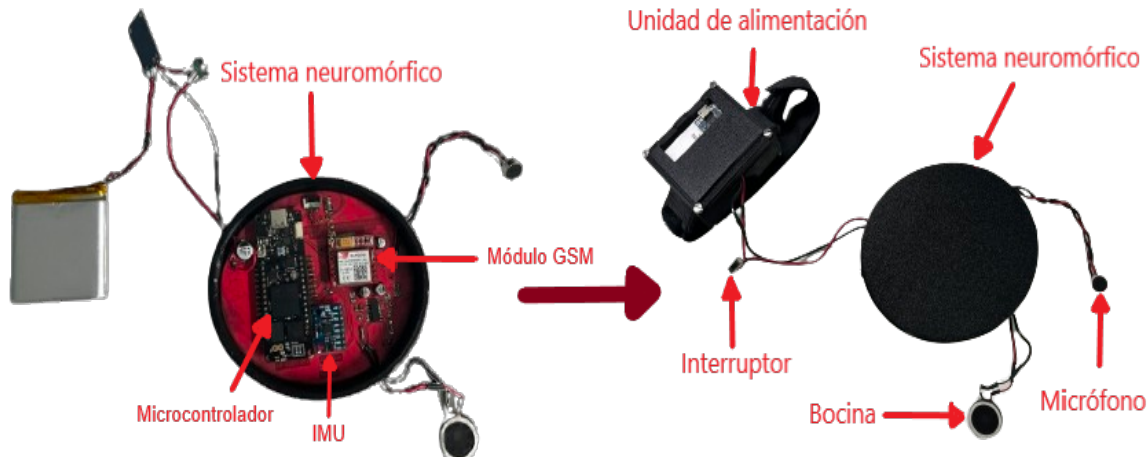


Figura 3.44: Integración del sistema neuromórfico en enclosure para herramienta de asistencia wearable.

3.6.4 Pruebas y validación

Con el propósito de evaluar la eficiencia del sistema neuromórfico, se monitoreo el estado de la Unidad de Alimentación completamente cargada, para evaluar el consumo de cada uno de los componentes y analizar la duración del sistema encendido.

La duración estimada de la batería se calculó con una celda Li-Ion de 3.7 V y 2500 mAh, utilizando la siguiente expresión general:

$$t = \frac{C_{bat}}{I_{total}} \quad (3.21)$$

donde:

- t es el tiempo de autonomía en horas,
- C_{bat} es la capacidad de la batería en mAh,
- I_{total} es la corriente total consumida por el sistema (mA).

Para cada escenario se aplicó la ecuación (3.21), por ejemplo:

$$t_{monitoreo} = \frac{2500 \text{ mAh}}{104 \text{ mA}} = 24.03 \text{ h} \quad (3.22)$$

$$t_{activo} = \frac{2500 \text{ mAh}}{734 \text{ mA}} = 3.41 \text{ h} \quad (3.23)$$

$$t_{llamada} = \frac{2500 \text{ mAh}}{1034 \text{ mA}} = 2.42 \text{ h} \quad (3.24)$$

$$t_{pico} = \frac{2500 \text{ mAh}}{2234 \text{ mA}} = 1.12 \text{ h} \quad (3.25)$$

Se debe considerar que los valores son aproximados, ya que factores como la eficiencia de los convertidores DC-DC, la temperatura, el envejecimiento de la batería y la intensidad de las ráfagas de transmisión del módulo GSM pueden afectar significativamente la autonomía real del sistema.

Con la descripción detallada de los elementos de hardware, los procedimientos de adquisición de datos y el diseño del modelo de inferencia, este capítulo establece las bases metodológicas necesarias para la implementación del sistema neuromórfico. A partir de ello, en el siguiente capítulo se presentan los resultados experimentales obtenidos, así como el análisis del desempeño de los modelos evaluados durante las pruebas.

CAPÍTULO 4: PRESENTACIÓN Y DISCUSIÓN DE LOS RESULTADOS

En este capítulo se presentan los resultados obtenidos durante la etapa de implementación y validación del sistema propuesto. En primer lugar, se muestra un análisis gráfico de la respuesta de la unidad de medición inercial ante un evento crítico, como lo es, en este caso, una caída.

Se realiza un análisis entre los datos iniciales y los datos postprocesados para la creación y entrenamiento del modelo de Redes Nueronales.

Posteriormente, se muestra la estructura final del modelo de red neuronal desarrollado, incluyendo el número de capas densas y ocultas, el número de neuronas por capa, así como el total de parámetros entrenables. Esta información se presenta en forma de tabla, junto con el puntaje obtenido durante la fase de evaluación.

Posteriormente, se analizan los resultados de desempeño del modelo de inteligencia artificial mediante la visualización de una matriz de confusión, la cual permite evaluar el comportamiento del modelo en la fase de prueba. Además, se incluyen las gráficas de precisión (accuracy) y pérdida (loss) obtenidas durante las etapas de entrenamiento y validación, lo que permite analizar el aprendizaje del modelo y su capacidad de generalización.

Con el objetivo de explorar la separación y agrupamiento de las diferentes clases, se emplearon las técnicas de reducción de dimensionalidad t-SNE y UMAP, cuyos resultados se presentan en gráficos de dispersión que permiten visualizar la distribución de las clases en el espacio latente del modelo.

Dichos resultados se contrastan contra los resultados del modelo generado mediante la selección de características más importantes por XGBoost.

Resaltando las ventajas y desventajas de cada uno y su viabilidad para implementación en un sistema embebido para inferencia en tiempo real.

A su vez, se analizan los mapas de calor generados por los algoritmos de índices de correlación de Pearson de cada categoría de movimiento inercial seleccionada.

De la misma manera, se presenta el análisis de los tiempos de respuesta del sistema, así como el análisis del consumo y la eficiencia energética del mismo.

Finalmente, se presentan los resultados de inferencia en tiempo real obtenidos a partir de la herramienta de asistencia desarrollada en este trabajo de investigación, evaluando su desempeño en diferentes pruebas experimentales para validar la funcionalidad y aplicabilidad del sistema propuesto.

4.1 Gráficos de registro de actividades inerciales

Se realizó una serie de pruebas preliminares para determinar la configuración óptima para obtener datos cruciales para la determinación de patrones de cada uno de los movimientos estipulados, descartando posibles saturaciones del sensor para tener la mayor confiabilidad posible en nuestra aplicación de Inteligencia Artificial para la identificación de patrones.

Se obtuvo la configuración adecuada para el sensor de movimiento inercial en 8g para el acelerómetro y en 1000°/s para el giroscopio, tomando en cuenta las presentadas en [Tabla 3.2](#) y [Tabla 3.3](#)

A continuación presentamos como es la respuesta típica del registro de una categoría (caída) mediante la UMI. En la [Figura 4.2](#) se muestran los detalles de la respuesta inicial para aceleración y velocidad angular obtenidos a partir de las 100 muestras iniciales (equivalentes a 800 datos). Estas gráficas en formato de líneas suavizadas con marcadores permiten apreciar la evolución temporal completa de las magnitudes medidas por la Unidad de Medición Inercial, incluyendo tanto los periodos de reposo como las fases previas, durante y posteriores al impacto.

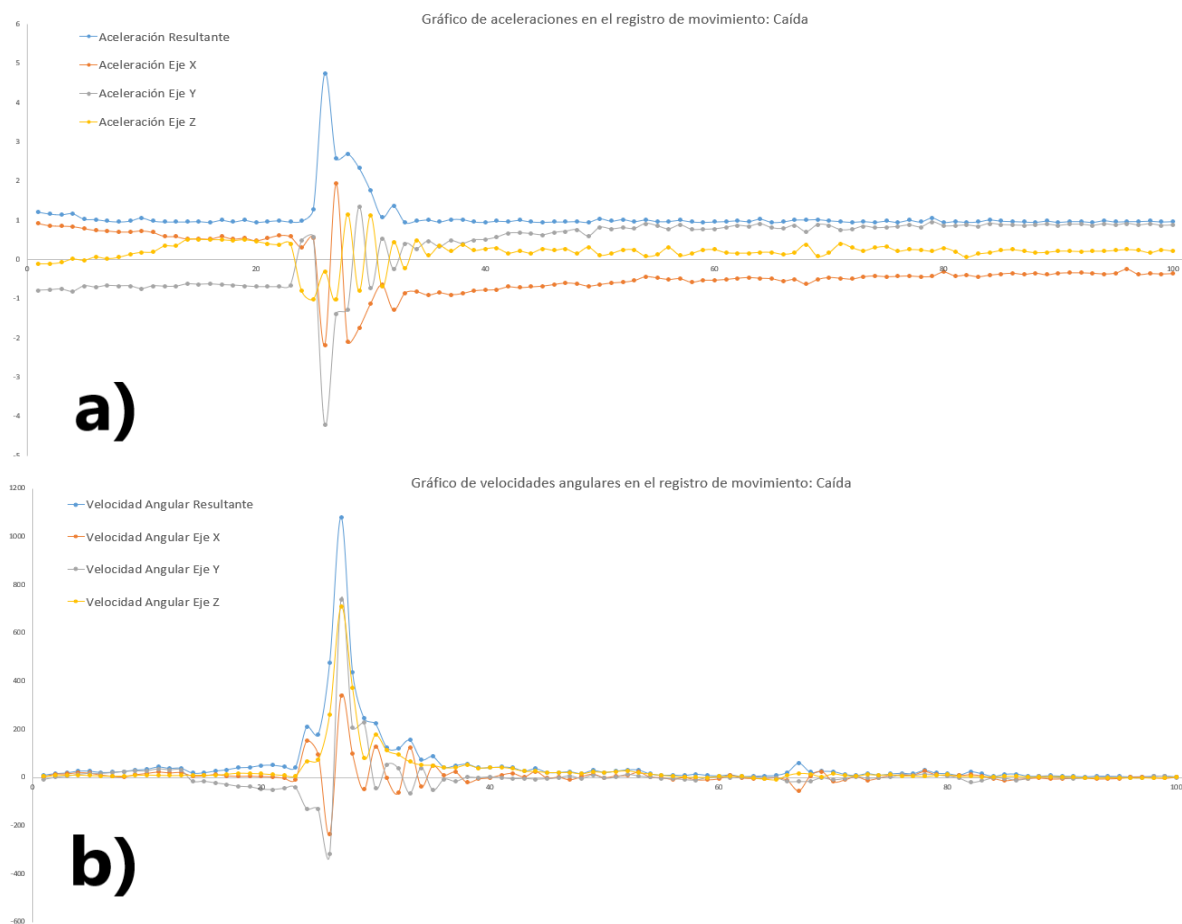


Figura 4.1: Registro de un evento crítico Caída "mediante la UMI. a) Datos iniciales de aceleración b) Datos iniciales de velocidad angular

En las gráficas se pueden identificar variaciones abruptas en cada uno de los ejes (x,y,z) respecto a la línea base, mientras que la resultante se caracteriza por que su excursión presenta una especie de hombros.

Estos datos poseen información clave para inferir de manera automática el evento al que nos estamos refiriendo (Caída); sin embargo, la densidad de los datos impide que dicho proceso automático se pueda realizar en tiempo real mediante un sistema microcontrolador convencional. Esta situación se solucionó mediante una estrategia de selección de características representativas a través del uso del algoritmo de Cálculo de energía por sumatoria del valor absoluto y XGBoost.

A continuación, en la [Figura 4.2](#) se muestra como es la curva de respuesta para los registros inerciales reducidos de las gráficas anteriores.

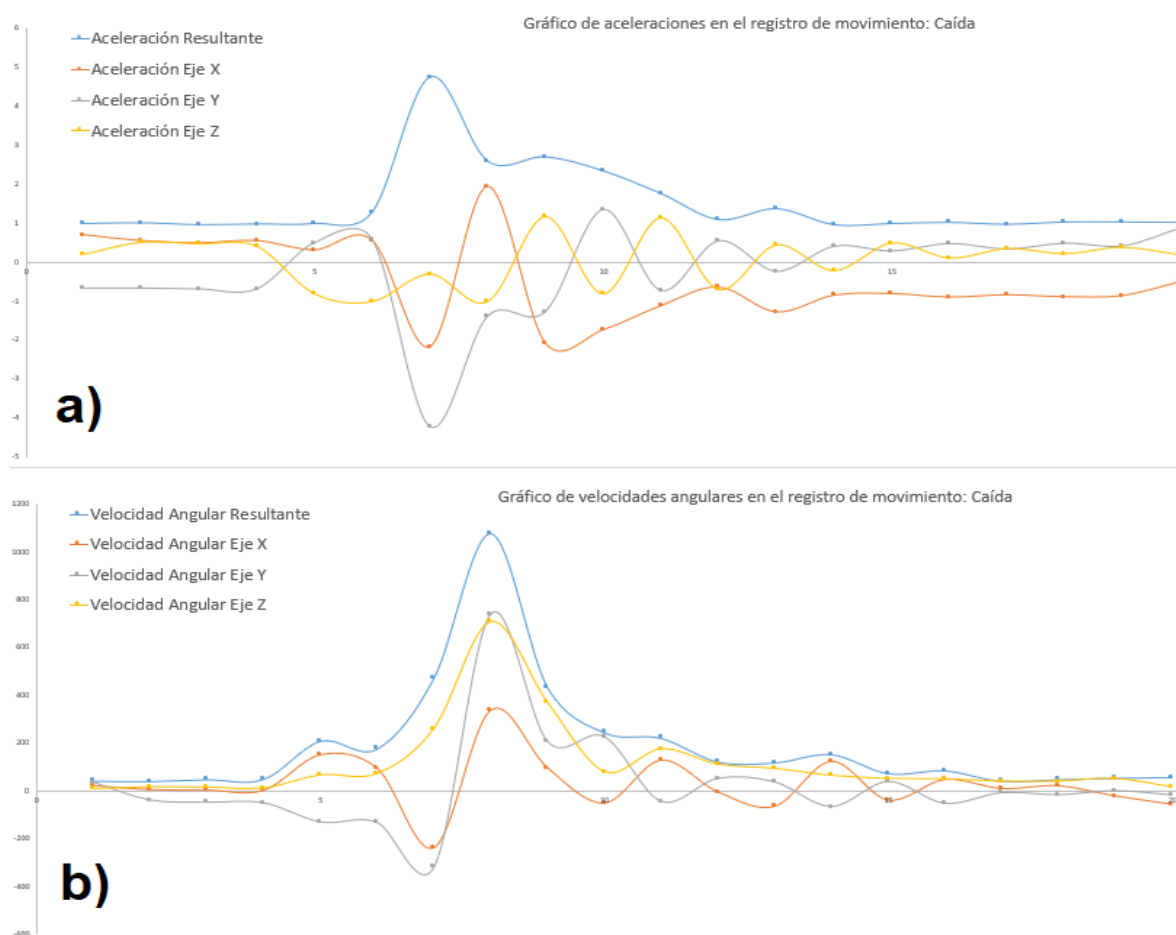


Figura 4.2: Registro de un evento crítico Caída "mediante la UMI. a) Datos reducidos de aceleración b) Datos reducidos de velocidad angular

Este nuevo formato se compone ahora de 80 datos por sensor (aceleración y velocidad angular), para un total de 160 datos, los cuáles son susceptibles de ser procesados en tiempo real mediante un sistema automático de inferencia por IA en un microcontrolador de gama media, tal como el que en este trabajo hemos propuesto (Portenta H7 Lite).

El proceso anterior se aplicó de manera similar a cada una de las clases de la [Tabla 3.5](#). De este modo, se garantizó que todas las categorías de actividad, tales como caminar, sentarse, levantarse y agacharse, fueran sometidas al mismo procedimiento de reducción y extracción

de información relevante. Esta metodología permitió conformar una base de datos en la cual cada clase conserva las características distintivas de su dinámica inercial.

4.2 Modelo de Red Neuronal Multicapa por sumatoria de valores absolutos

Como uno de los resultados principales y fundamentales para el desarrollo e implementación del sistema neuromórfico, a continuación, en la [Tabla 4.1](#) se presenta la estructura final del modelo de red neuronal implementado en el microcontrolador del sistema neuromórfico.

Tabla 4.1: Estructura final del modelo de red neuronal

Layer (type)	Output Shape	Param #
dense_4 (Dense)	(None, 256)	40,704
leaky_re_lu_3 (LeakyReLU)	(None, 256)	0
dropout_2 (Dropout)	(None, 256)	0
dense_5 (Dense)	(None, 128)	32,896
leaky_re_lu_4 (LeakyReLU)	(None, 128)	0
dropout_3 (Dropout)	(None, 128)	0
dense_6 (Dense)	(None, 64)	8,256
leaky_re_lu_5 (LeakyReLU)	(None, 64)	0
dense_7 (Dense)	(None, 11)	715

Total params: 82,571 Trainable params: 82,571 Non-trainable params: 0

Como se puede analizar en la [Tabla 4.1](#), el modelo está conformado por 4 capas densas, siendo 11 las categorías que ingresan al modelo. La primera capa consiste en una capa de 256 neuronas y 40,704 parámetros, con una función de activación Leaky ReLU; posteriormente, se integra una capa densa de 128 neuronas y 40,704 parámetros. Después, se integra una capa densa de 64 neuronas y 8,256 parámetros, dando como salida a la última capa, que es la capa de salidas y contiene 11 neuronas y 715 parámetros.

De este modo, tenemos un modelo de Red Neuronal con 82,571 parámetros entrenables.

A continuación se muestra en la [Tabla 4.2](#) el análisis de precisión, recall, F1-score y support de cada una de las clases durante la fase de prueba.

Tabla 4.2: Reporte de clasificación del modelo en la fase de validación

Clase	Precision	Recall	F1-score	Support
agacharse	1.00	1.00	1.00	10
caida_lateralD	1.00	1.00	1.00	10
caminar_lento	1.00	1.00	1.00	10
caminar_normal	1.00	1.00	1.00	10
caminar_rapido	1.00	1.00	1.00	10
levantarse_lento	0.83	1.00	0.91	10
levantarse_normal	1.00	0.71	0.83	14
levantarse_rapido	0.83	1.00	0.91	10
sentarse_lento	1.00	1.00	1.00	10
sentarse_normal	1.00	1.00	1.00	10
sentarse_rapido	1.00	1.00	1.00	10
Accuracy	0.96			114
Macro avg	0.97	0.97	0.97	114
Weighted avg	0.97	0.96	0.96	114

En la [Tabla 4.2](#) se presentan las métricas de desempeño del modelo en la fase de prueba, considerando las métricas de precisión, Recall y F1-score para cada una de las once clases analizadas. En términos generales, el modelo alcanzó un valor de Accuracy (Exactitud) global del 96 %, con valores de macro average y weighted average de 0.97 y 0.96, respectivamente. Estos resultados son bastante aceptables para el tipo de aplicación que se busca.

La [Tabla 4.2](#) permite observar que la mayoría de las actividades presentan valores superiores a 0.9 en precisión, recall y F1-score (1.00). Con este resultado, podemos afirmar que las actividades tales como agacharse, caida_lateralD, caminar (en sus tres variaciones de velocidad) y sentarse (en sus tres variaciones de velocidad), son las clases que presentan una predicción

ideal.

Sin embargo, se identifican ligeras disminuciones en las métricas para las clases `levantarse_lento`, `levantarse_normal` y `levantarse_rapido`. En el caso de `levantarse_normal`, el Recall es de 0.71, lo que sugiere que el modelo no logró identificar correctamente algunos ejemplos pertenecientes a esta clase, clasificándolos posiblemente como otra variación de la acción de levantarse. A pesar de ello, su Precision se mantuvo en 1.00, lo que indica que las predicciones realizadas para esta clase fueron correctas. Los valores de precisión de 0.83 se pueden interpretar como la dificultad que tiene el modelo para distinguir sutiles variaciones ante movimientos que comparten patrones cinemáticos.

Finalmente, los resultados de la tabla anterior respaldan la viabilidad del modelo seleccionado como herramienta confiable para clasificar actividades humanas (clases), incluyendo la detección precisa de caídas.

A continuación se muestra en [Figura 4.3](#) un gráfico representativo del porcentaje de pérdida en entrenamiento y validación del modelo.

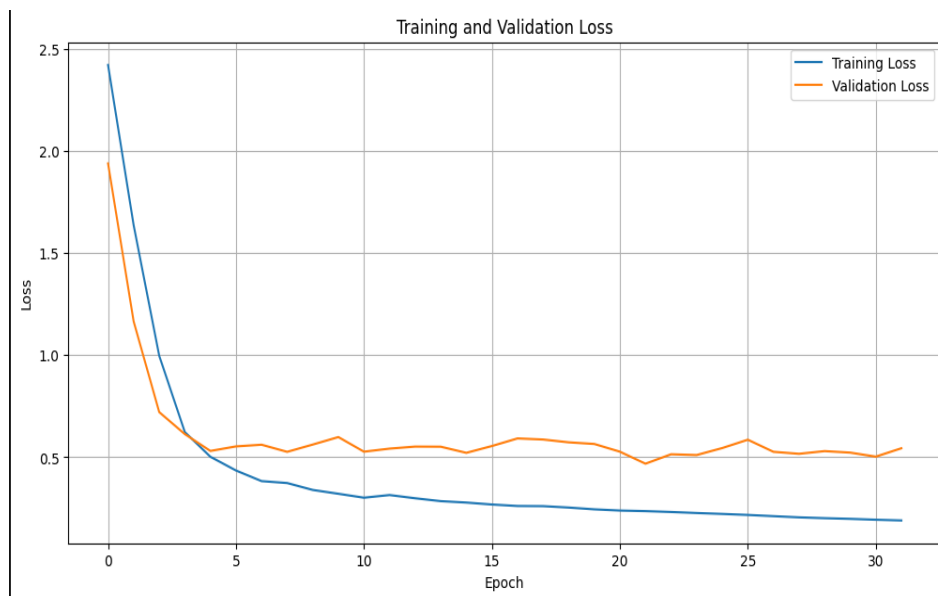


Figura 4.3: Gráfico representativo de pérdida en la fase de entrenamiento y fase de validación del modelo.

En la [Figura 4.3](#) se observa el comportamiento de la función de pérdida (loss) durante el

proceso de entrenamiento y validación del modelo. El trazo azul representa la pérdida en el conjunto de entrenamiento, mientras que el trazo naranja corresponde a la pérdida en el conjunto de validación.

Al inicio del entrenamiento, ambas pérdidas presentan valores elevados, siendo la pérdida de entrenamiento aproximadamente 2.4 y la de validación alrededor de 1.9. Sin embargo, durante las primeras cinco épocas, se aprecia una disminución pronunciada en ambas curvas, lo que indica que el modelo está aprendiendo de manera adecuada y ajustando sus parámetros manera eficiente.

A partir de la época 5, la curva de validación comienza a estabilizarse alrededor de 0.5, mientras que la curva de entrenamiento continúa descendiendo de manera gradual hasta alcanzar un valor cercano a 0.2. Esta diferencia entre ambas regiones de las gráficas sugiere la presencia de un ligero sobreajuste (overfitting), dado que el modelo sigue mejorando con los datos de entrenamiento.

No obstante, la diferencia entre ambas pérdidas no es excesiva, lo que indica que el modelo mantiene una buena capacidad de generalización. La estabilización temprana de la curva de validación también sugiere que el uso de técnicas como early stopping fue efectivo para evitar un sobreentrenamiento.

En conclusión, este comportamiento refleja que el modelo logra un buen equilibrio entre aprendizaje y generalización, siendo capaz de adaptarse a los datos de entrenamiento sin perder su capacidad de predecir adecuadamente en datos no registrados.

Como continuación y complementación del desempeño del modelo de Red Neuronal, a continuación en la [Figura 4.4](#) se muestra un gráfico de la matriz de confusión del modelo evaluado en la fase de validación.

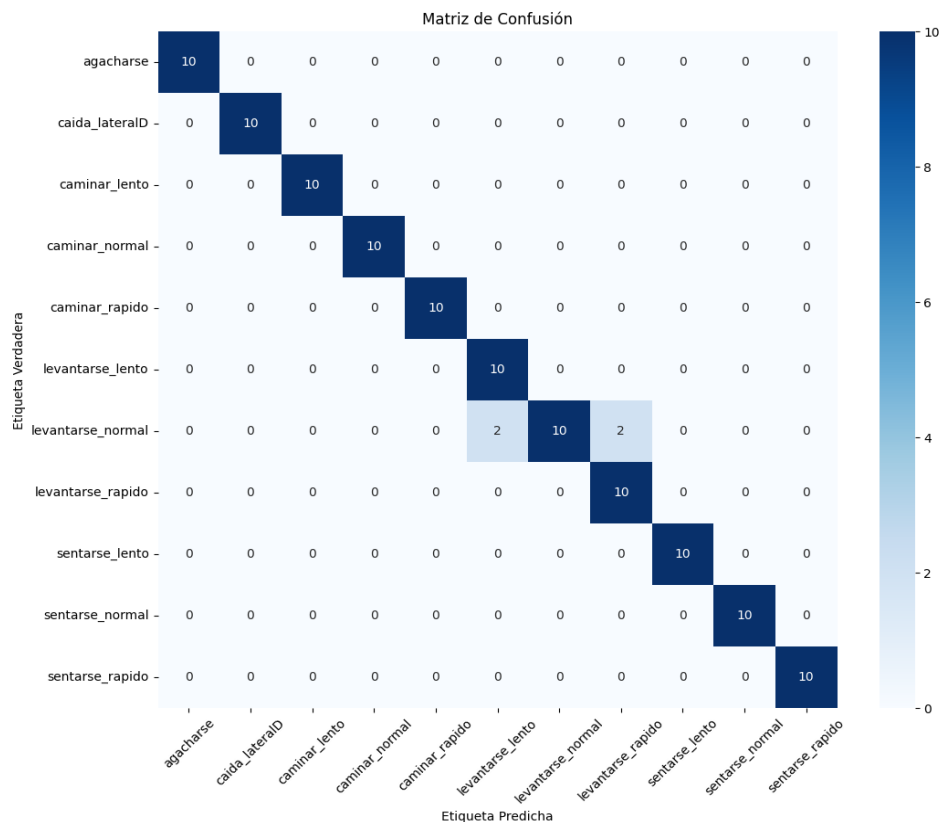


Figura 4.4: Matriz de confusión para validación del modelo de Red Neuronal.

En la [Figura 4.4](#) se analiza el desempeño en la clasificación de las 11 actividades consideradas en este estudio. Cada fila representa la etiqueta verdadera, mientras que cada columna corresponde a la etiqueta predicha por el modelo.

Se observa que, para la mayoría de las clases, el modelo logra una clasificación perfecta con 10 aciertos de 10 posibles. Esto es evidente en las actividades: agacharse, caída lateral derecha, caminar lento, caminar normal, caminar rápido, levantarse lento, levantarse rápido, sentarse lento, sentarse normal y sentarse rápido, donde todos los ejemplos se encuentran correctamente ubicados en la diagonal principal.

El único caso con errores se presenta en la actividad "levantarse normal", donde de los 14 ejemplos evaluados, 10 fueron clasificados correctamente, mientras que 4 fueron mal clasificados: dos como levantarse lento y dos como levantarse rápido. Este comportamiento sugiere que, al realizar el movimiento de levantarse a una velocidad normal, el patrón de datos

puede confundirse con los generados a velocidades lentas y rápidas.

En términos generales, la matriz refleja un desempeño sobresaliente, evidenciado por la fuerte concentración de predicciones en la diagonal principal y una precisión global superior al 96 %. Estos resultados indican que el modelo ha aprendido a distinguir con eficacia la mayoría de los eventos, siendo adecuado para identificar caídas y movimientos cotidianos.

Otro resultado importante es el que se refiere a la dispersión de las clases evaluadas por el modelo. A continuación, se presenta el gráfico.

Con el objetivo de visualizar la distribución de las características obtenidas por el modelo y evaluar la separabilidad entre las clases, se empleó la técnica de reducción de dimensionalidad t-distributed Stochastic Neighbor Embedding (t-SNE). Esta técnica es ampliamente utilizada para representar datos de alta dimensión en un espacio bidimensional, preservando la estructura local y permitiendo identificar patrones y agrupamientos.

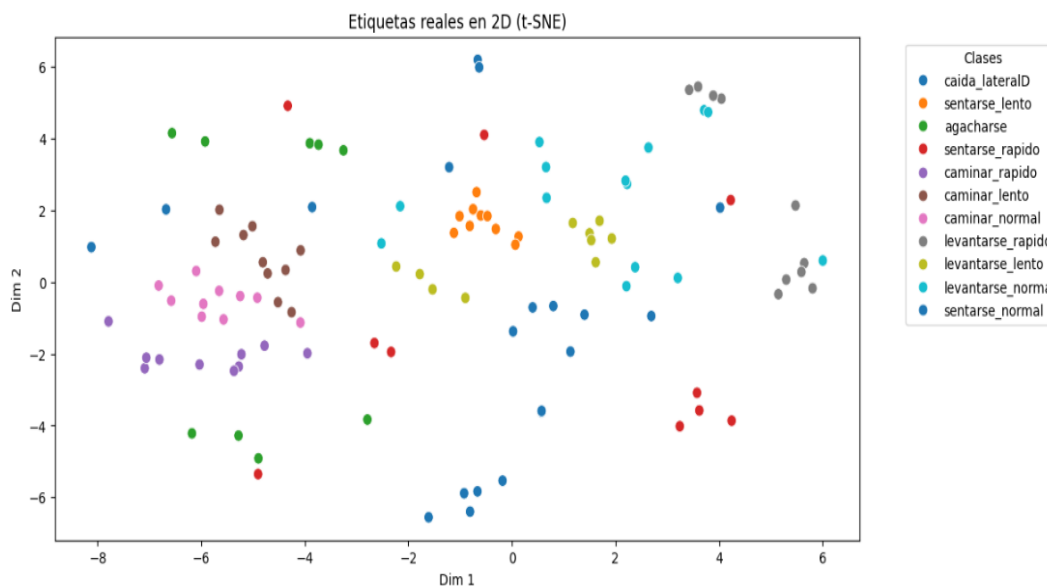


Figura 4.5: Gráfico de dispersión de las mediciones reales en la fase de prueba.

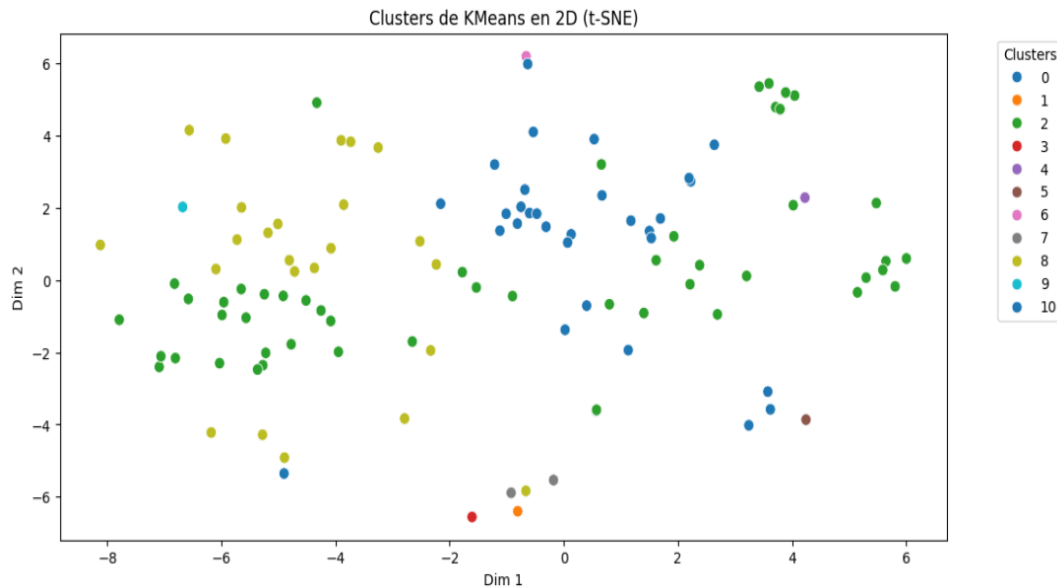


Figura 4.6: Gráfico de dispersión del Clustering por K-means en la fase de prueba.

En la [Figura 4.5](#), los puntos representan las mediciones de la fase de prueba, coloreados según su etiqueta real. Esta representación permite observar qué tan bien diferenciadas se encuentran las actividades en el espacio de características. Una separación clara entre grupos indicaría que el modelo cuenta con información suficiente para discriminar entre las distintas clases, mientras que la superposición parcial sugeriría similitud en los patrones de movimiento.

Posteriormente, en la [Figura 4.6](#) se aplicó el algoritmo de K-Means con el mismo número de clústeres que las clases. Este gráfico permite analizar cómo un método de agrupamiento no supervisado organiza los datos únicamente en función de sus características, sin información previa sobre las etiquetas. Al comparar ambos gráficos, es posible evaluar si las clases reales tienen una correspondencia directa con los clústeres generados, lo que sirve como un indicador adicional de la coherencia y la calidad de los datos empleados en la clasificación.

Esta visualización complementa el análisis cuantitativo presentado anteriormente, ofreciendo una perspectiva visual del rendimiento del modelo y del grado de diferenciación entre las actividades analizadas.

4.3 Correlación de variables

Una vez visualizada la distribución espacial de las mediciones mediante t-SNE y analizado el grado de separación entre clases, resulta pertinente complementar este análisis explorando la relación interna entre las características que conforman cada actividad. Mientras que el t-SNE se enfoca en la representación global de los datos en un espacio bidimensional, el análisis de correlaciones permite examinar cómo se comportan las variables entre sí dentro de cada clase, proporcionando información más detallada sobre la estructura y la redundancia de los datos.

Para este propósito, se emplearon mapas de calor basados en el índice de correlación de Pearson, calculados de manera independiente para cada conjunto de mediciones agrupadas por clase. Este índice cuantifica la fuerza y dirección de la relación lineal entre dos variables, tomando valores entre -1 y 1. Un valor cercano a 1 indica una correlación positiva fuerte (ambas variables aumentan o disminuyen juntas); un valor cercano a -1 señala una correlación negativa fuerte (una variable aumenta mientras que la otra disminuye), y valores cercanos a 0 reflejan una relación lineal baja o nula.

El uso de mapas de calor facilita la identificación visual de patrones, permitiendo detectar grupos de características altamente correlacionadas que podrían aportar información similar al modelo. Este análisis es especialmente útil para comprender el comportamiento de los sensores durante cada actividad, identificar posibles redundancias en las mediciones y evaluar la calidad de las características empleadas en el entrenamiento.

De esta forma, el estudio de correlaciones se presenta como un paso complementario al análisis previo con t-SNE: mientras que este último evidenció la separación entre clases en el espacio de características, los mapas de calor permiten profundizar en la estructura interna de los datos, aportando una perspectiva más detallada, utilizada tanto para la optimización de la base de datos como para la mejora del rendimiento del modelo de clasificación.

Tomando en cuenta el orden de clases presentado en el modelo, como se muestra en la [Tabla 4.2](#), a continuación, en la [Figura 4.7](#) se presenta la matriz de correlación de Pearson de variables para la base de datos de la clase Agacharse.

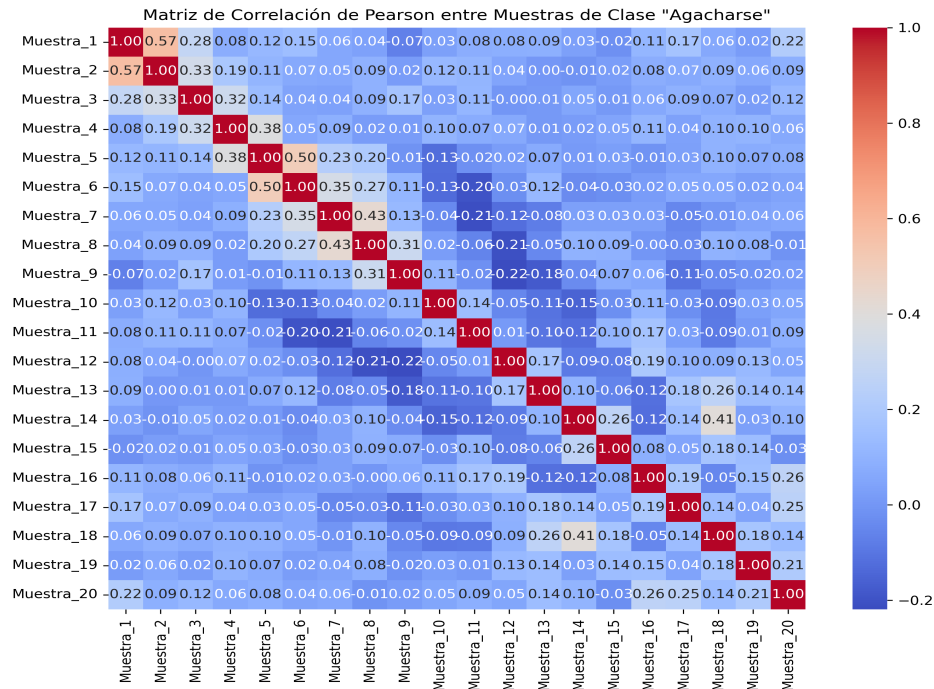


Figura 4.7: Gráfico de Índices de Correlación de Pearson para la clase Agacharse.

Si analizamos el gráfico anterior, podemos ver que las variables por muestra de cada bloque de 8 datos inerciales (muestra) tienen un índice de correlación inferior a 0.50 en general, con excepción de la muestra 2 respecto a la muestra 1, con un valor de 0.56, y la muestra 6 respecto a la muestra 5, con un valor de 0.50. Este nivel de correlación sugiere que las mediciones de los sensores (acelerómetro y giroscopio) están captando aspectos distintos del movimiento, lo cual es ideal para el modelo, pues enriquece la información sin que las variables sean duplicadas.

Continuando con el orden, en la [Figura 4.8](#) se presenta la matriz de correlación de las variables de la clase Caída, siendo una de las clases fundamentales para el sistema neuromórfico embebido, que permite realizar la inferencia y alerta en la herramienta de asistencia enfocada en los adultos mayores.

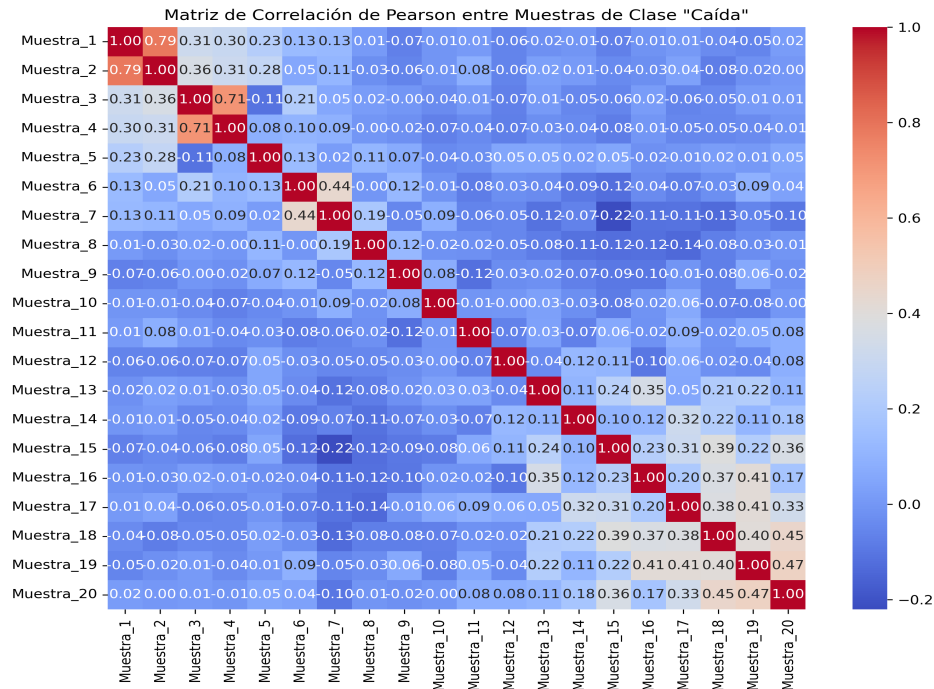


Figura 4.8: Gráfico de Índices de Correlación de Pearson para la clase Caída.

Similar a la clase Agacharse, la clase Caída presenta un índice de correlación aceptable entre las variables en general, con excepción de la muestra 2 respecto a la muestra 1, que tiene un valor de 0.79, y de la muestra 4 respecto a la muestra 3, que tiene un valor de 0.71. Las muestras restantes presentan una correlación positiva menor a 0.50, lo cual indica una distribución de datos que aporta información adecuada para el modelo. A través de esta clase, se evaluará la inferencia mediante el microcontrolador para notificar o no el evento de emergencia a través de una llamada o un mensaje de texto.

Pasando ahora a las clases con subcategorías, analizamos la matriz de correlación de variables para la clase Caminar Lento en la [Figura 4.9](#), la cual, a diferencia de las matrices anteriores, presenta índices de correlación positivos más altos, lo que indica una similitud entre las muestras durante las mediciones.

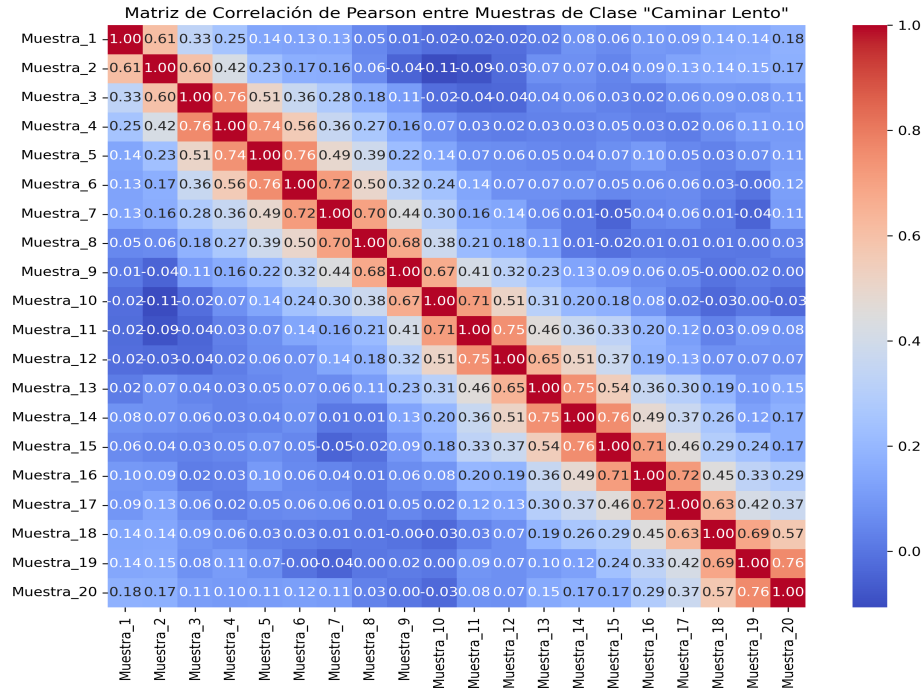


Figura 4.9: Gráfico de Índices de Correlación de Pearson para la clase Caminar Lento.

Como se puede analizar en la [Figura 4.9](#), existe una similitud entre las variables que es mucho mayor que en las clases analizadas anteriormente; sin embargo, esto es esperado debido a la naturaleza del movimiento. Siendo que este movimiento se registró con una velocidad lenta, las variaciones de aceleración y velocidad presentan un cambio pequeño en el lapso de tiempo de registro y, en consecuencia, el índice de correlación entre las variables es más alto. En casos como este, se considera recomendable hacer ajustes en la base de datos de dicha clase, con el propósito de mejorar la precisión en caso de que el modelo presente errores en la predicción de la clase durante la fase de validación.

A continuación, en la siguiente [Figura 4.10](#) se muestra el gráfico de correlación para la clase Caminar Normal, el cual presenta una mejora en la correlación entre variables respecto a su variante anterior.

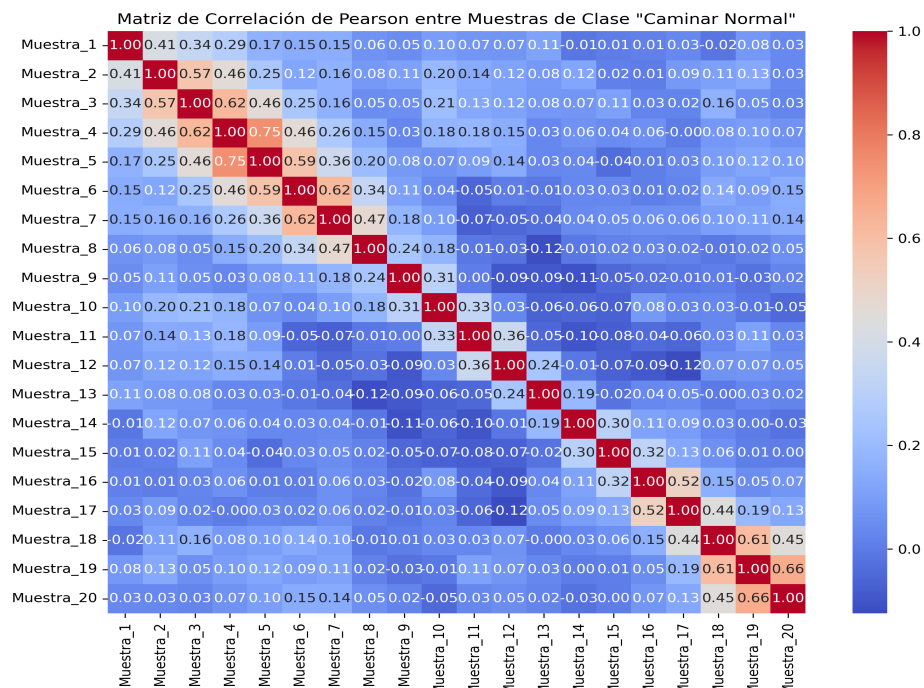


Figura 4.10: Gráfico de Indices de Correlación de Pearson para la clase Caminar Normal.

Al analizar la figura anterior, se observa que la clase Caminar Normal presenta una correlación positiva menor en comparación con la clase previa. Esto es coherente con el comportamiento esperado en datos inerciales, dado que los patrones de movimiento dependen de factores como la naturaleza de la actividad realizada, la velocidad de ejecución y la sensibilidad de la unidad de medición inercial (IMU, Inertial Measurement Unit).

En esta clase, la matriz de correlación evidencia una similitud significativa entre las variables 3 y 7, con coeficientes superiores a 0.5. Este nivel de correlación puede atribuirse, en parte, a la metodología empleada para el registro de datos. Debido a que los movimientos no se registraron de manera continua, cada medición inicia desde un punto común dentro del ciclo de movimiento. Esto genera patrones iniciales muy similares entre las distintas mediciones de la clase. Sin embargo, conforme avanza el movimiento, las demás variables presentan una mayor variabilidad, ya que resulta complejo obtener exactamente la misma aceleración y velocidad angular en cada registro, incluso cuando se controlan la velocidad y la orientación durante la ejecución del movimiento.

Finalizando con las subclases del movimiento Caminar, concluimos con el gráfico de correlación de la clase Caminar Rápido, que se presenta a continuación en la [Figura 4.11](#).

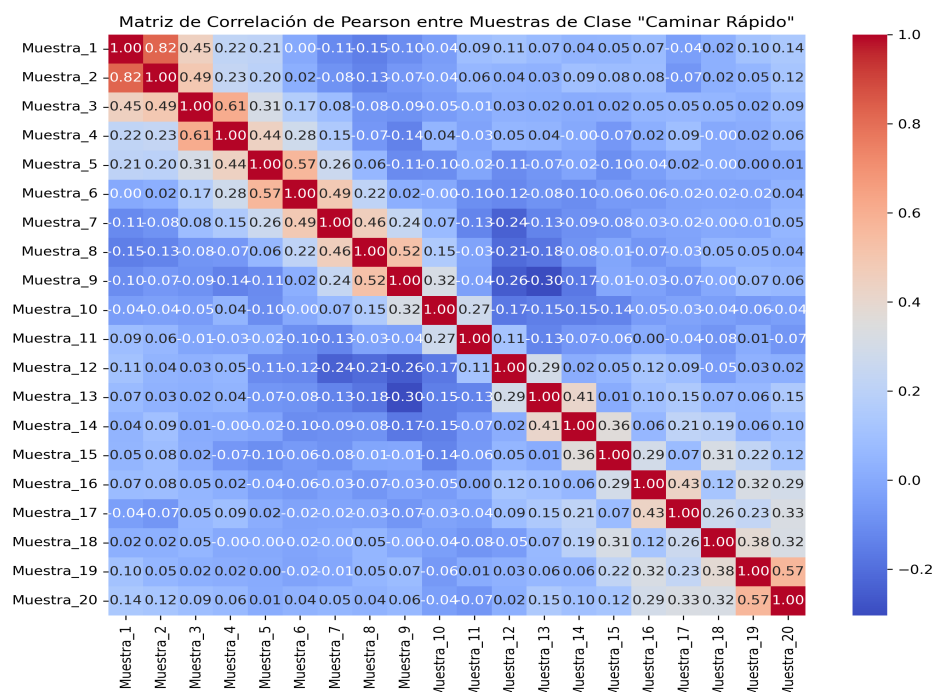


Figura 4.11: Gráfico de Índices de Correlación de Pearson para la clase Caminar Rápido.

Como se puede observar en la figura anterior, la correlación positiva de esta clase es deseable, ya que, al igual que en la clase anterior, se presentan índices de correlación superiores a 0.50 únicamente entre las muestras 3 y 7, debido a la naturaleza y el registro del movimiento.

Como se verá a continuación en las siguientes figuras, el movimiento correspondiente a Levantarse de un asiento presenta índices de correlación positivos demasiado altos y en un rango de muestras indeseable; sin embargo, es por ello que en una de las clases pertenecientes a este movimiento se realizó un ajuste de mediciones en la Base de Datos, con el objetivo de mejorar su precisión.

A continuación, en la [Figura 4.12](#) se presenta el gráfico de correlación de variables para la clase Levantarse Lento.

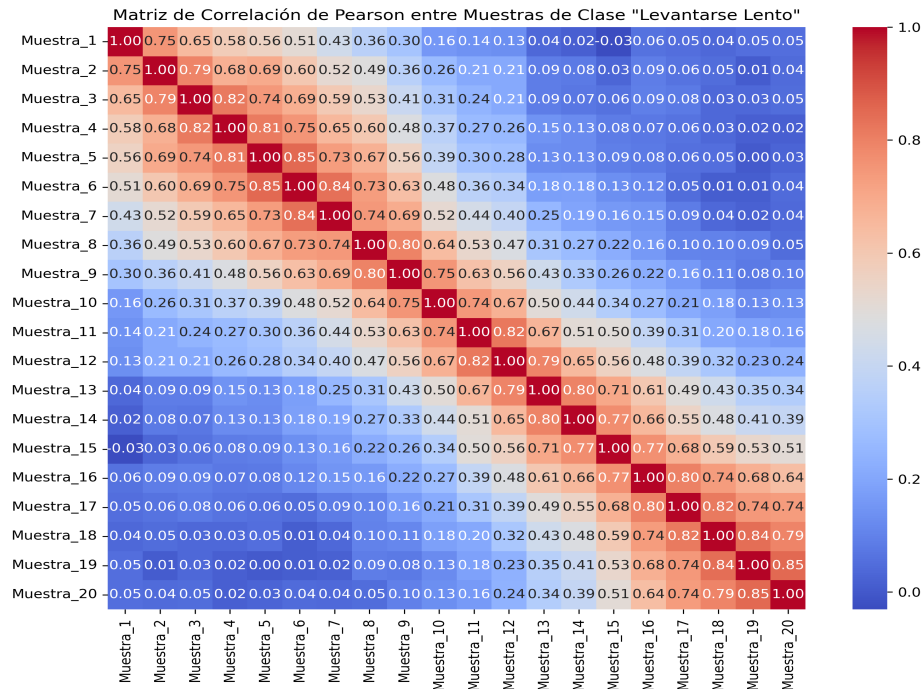


Figura 4.12: Gráfico de Índices de Correlación de Pearson para la clase Levantarse Lento.

La matriz de correlación para la clase Levantarse Lento evidencia índices significativamente más altos en comparación con otras clases, lo que indica una mayor similitud entre las variables registradas. Se observa que múltiples pares de muestras presentan coeficientes de correlación superiores a 0.7, y algunos incluso cercanos a 0.85, lo que sugiere que los movimientos registrados son consistentes y reproducibles entre diferentes mediciones. Este patrón de alta correlación puede explicarse por la naturaleza controlada y lenta del movimiento, donde la velocidad y la aceleración cambian de manera gradual, permitiendo que las variables del sensor inercial mantengan relaciones estables. Además, el inicio de cada registro parte de una posición similar, lo que contribuye a la homogeneidad observada en las mediciones. En conjunto, estos resultados reflejan que la clase Levantarse Lento tiene un patrón cinemático más uniforme, debido a la naturaleza del movimiento y la metodología del registro del mismo.

Como se observará a continuación en la [Figura 4.13](#), la clase Levantarse Normal refleja un alto índice de correlación entre variables, al igual que en la clase Levantarse Lento. Es debido a que aunque varíe la velocidad de ejecución del movimiento, la cinemática del movimiento, y

los datos registrados por los sensores tienen cierta similitud entre si.

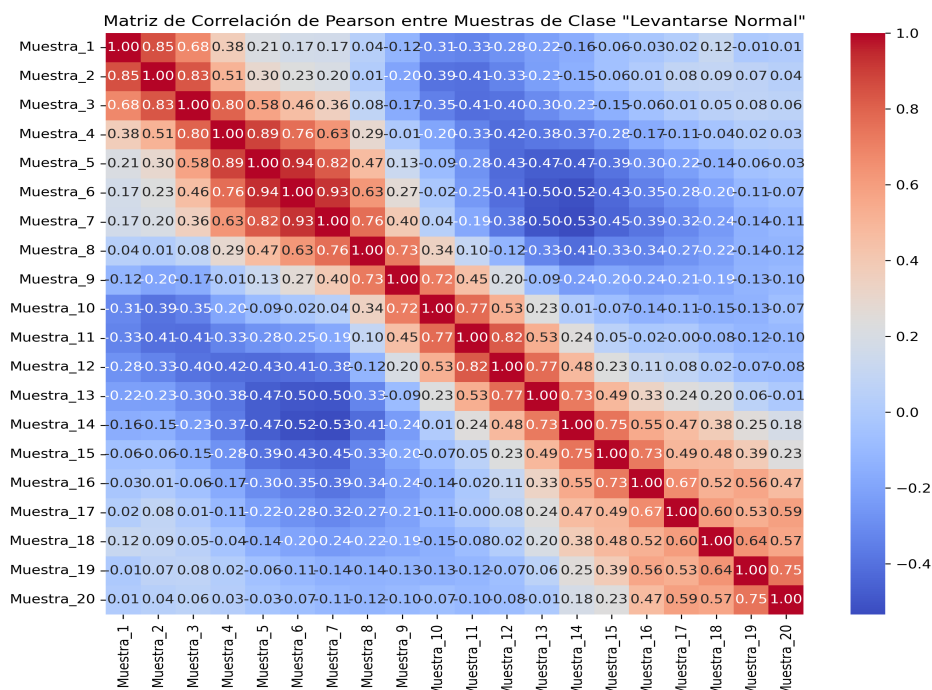


Figura 4.13: Gráfico de Índices de Correlación de Pearson para la clase Levantarse Normal.

La matriz de correlación para la clase Levantarse Normal muestra índices moderados en comparación con la clase Levantarse Lento, con valores de correlación que, en su mayoría, se encuentran en un rango de 0.3 a 0.8. Esto indica una alta similitud entre las variables, lo que sugiere una mayor variabilidad en la ejecución del movimiento. Este comportamiento puede atribuirse a que, al realizar la acción de levantarse a una velocidad normal, los cambios en la aceleración y la velocidad angular ocurren de forma más abrupta, lo que introduce diferencias en las mediciones, incluso cuando se controla el punto inicial del registro. Además, la menor homogeneidad entre las muestras refleja que factores como ligeras variaciones en la postura inicial, la orientación del cuerpo o el ritmo de ejecución afectan significativamente las señales capturadas por la unidad de medición inercial (IMU). En consecuencia, esta clase presenta patrones cinemáticos menos consistentes, lo que la hace más compleja de modelar y diferenciar en relación con movimientos similares.

Aunque la clase Levantarse Normal presenta una correlación alta entre sus muestras, esto únicamente refleja similitud interna en la ejecución del movimiento. Sin embargo, esta actividad comparte características cinemáticas con otras clases de transición postural, como Levantarse Lento o Sentarse Normal, lo que dificulta su separación en el espacio de características.

Es por ello que en la metodología de registro de datos se propuso registrar 20 mediciones adicionales para la clase Levantarse Normal, con el objetivo de mejorar la precisión al realizar las inferencias con nuevos datos de este movimiento, lo cual fue beneficioso para nuestro propósito, ya que se obtuvo una precisión para este movimiento del 90 %.

Finalizando con el movimiento de Levantarse, se presenta a continuación en la [Figura 4.14](#) la matriz de correlación de la clase Levantarse Rápido, la cual presenta una correlación positiva más alta que las clases anteriores, lo que sugiere una mayor variabilidad en la ejecución del movimiento y, por lo tanto, una mejor precisión en la inferencia de esta clase.

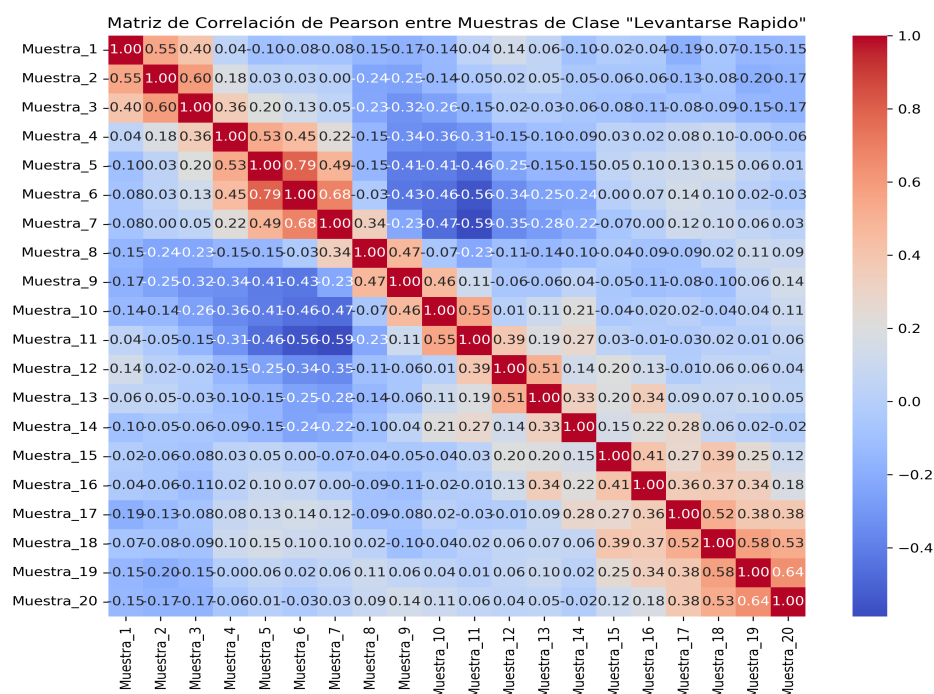


Figura 4.14: Gráfico de Índices de Correlación de Pearson para la clase Levantarse Rápido.

La matriz de correlación correspondiente a la clase Levantarse Rápido muestra, en su mayoría,

coeficientes de correlación bajos, con valores que se concentran entre 0.0 y 0.4, y solo algunos pares alcanzan índices cercanos a 0.7. Esto refleja una baja similitud entre las variables, indicando una elevada variabilidad en las mediciones. Este comportamiento se debe a la naturaleza del movimiento rápido, donde los cambios bruscos en aceleración y velocidad angular generan diferencias significativas entre cada ejecución, incluso si se controla la postura inicial y el inicio del registro. Además, a altas velocidades, factores como la ligera desincronización en el inicio del levantamiento, la dirección del impulso y las microvariaciones en la fuerza aplicada influyen directamente en la señal capturada por la IMU. En consecuencia, esta clase presenta patrones cinemáticos menos consistentes, lo que implica un mayor reto para el modelado y la clasificación, ya que la dispersión de los datos dificulta la identificación de relaciones lineales estables entre las variables.

Después de analizar las matrices de correlación correspondientes al movimiento de Levantarse en sus tres subclases (Lento, Normal y Rápido), se procede a presentar los resultados para el movimiento opuesto: Sentarse. Al igual que en el caso anterior, este movimiento se ha dividido en tres subclases con base en la velocidad de ejecución: Sentarse Lento, Sentarse Normal y Sentarse Rápido. El objetivo de este análisis es identificar el grado de similitud entre las mediciones dentro de cada subclase y comparar cómo la variabilidad en los datos se ve afectada por la velocidad del movimiento.

En primera instancia, se examina la clase Sentarse Lento, donde se espera observar patrones de correlación más altos debido a la naturaleza controlada y gradual de este tipo de ejecución, lo que permite obtener señales más consistentes y homogéneas en los registros inerciales.

A continuación, en la [Figura 4.15](#) se muestra la matriz de correlación para la clase Sentarse Lento.

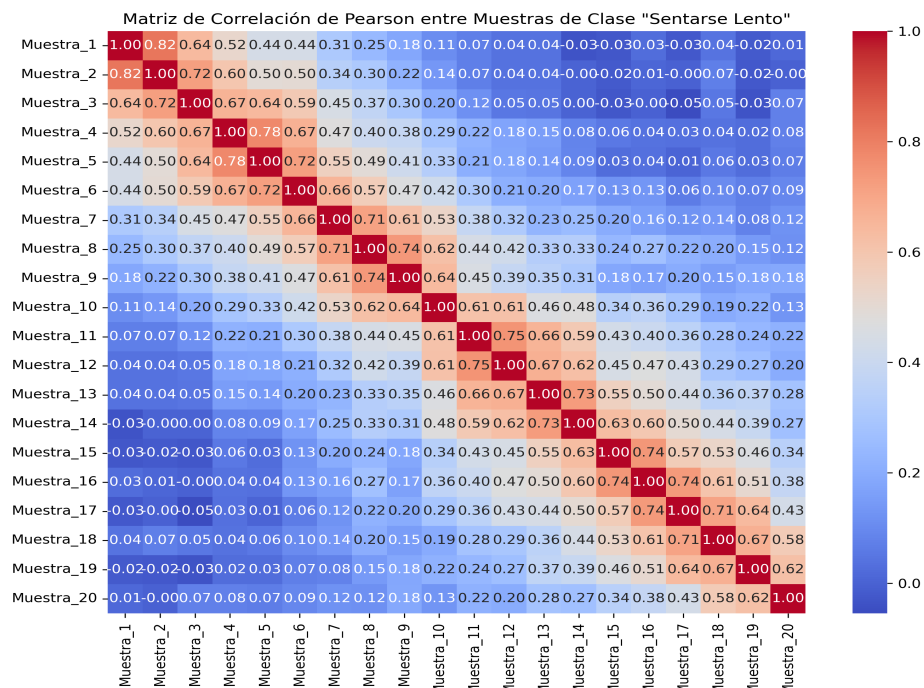


Figura 4.15: Gráfico de Indices de Correlación de Pearson para la clase Sentarse Lento.

La matriz de correlación correspondiente a la clase Sentarse Lento muestra valores que se distribuyen principalmente entre 0.2 y 0.7, con zonas aisladas que alcanzan valores superiores a 0.8 en algunas parejas de muestras. Esta distribución indica que la clase presenta un nivel moderado de similitud interna.

La variabilidad observada es coherente con la naturaleza del movimiento: al realizar la acción de sentarse lentamente, los usuarios tienden a modificar aspectos como el control del descenso, la inclinación del tronco, la orientación del torso y la velocidad angular de cada fase del movimiento. Estos cambios, aunque mantienen la estructura global del movimiento, generan diferencias apreciables en las mediciones capturadas por la IMU, lo que se refleja en correlaciones medias y una menor homogeneidad entre muestras.

Para continuar, en la [Figura 4.16](#) se presenta la matriz de correlación para la clase Sentarse Normal.

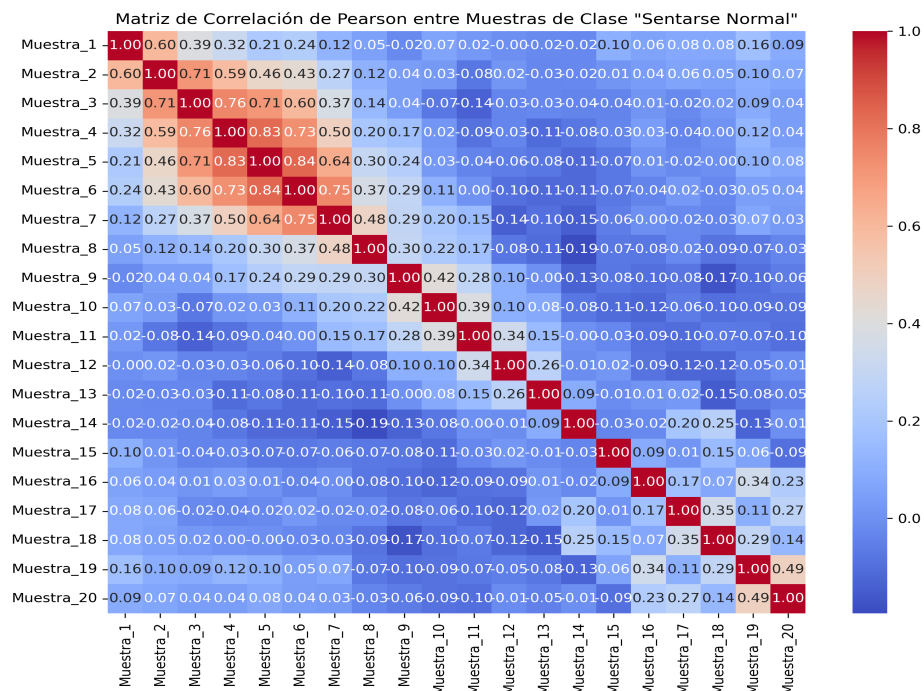


Figura 4.16: Gráfico de Índices de Correlación de Pearson para la clase Sentarse Normal.

La matriz de correlación correspondiente a la clase Sentarse Normal muestra valores que, en su mayoría, se encuentran entre 0.2 y 0.6, con algunas regiones localizadas que alcanzan correlaciones cercanas a 0.7. Este comportamiento indica que, aunque existen segmentos del movimiento con similitudes claras entre muestras, principalmente en las fases más controladas del gesto de sentarse. La clase presenta también zonas con correlaciones más bajas, asociadas a variaciones propias del patrón cinemático.

Para finalizar el análisis de correlación de variables entre cada una de las clases del modelo entrenado, se presenta a continuación en la [Figura 4.17](#) la matriz de correlación para la clase Sentarse Rápido.

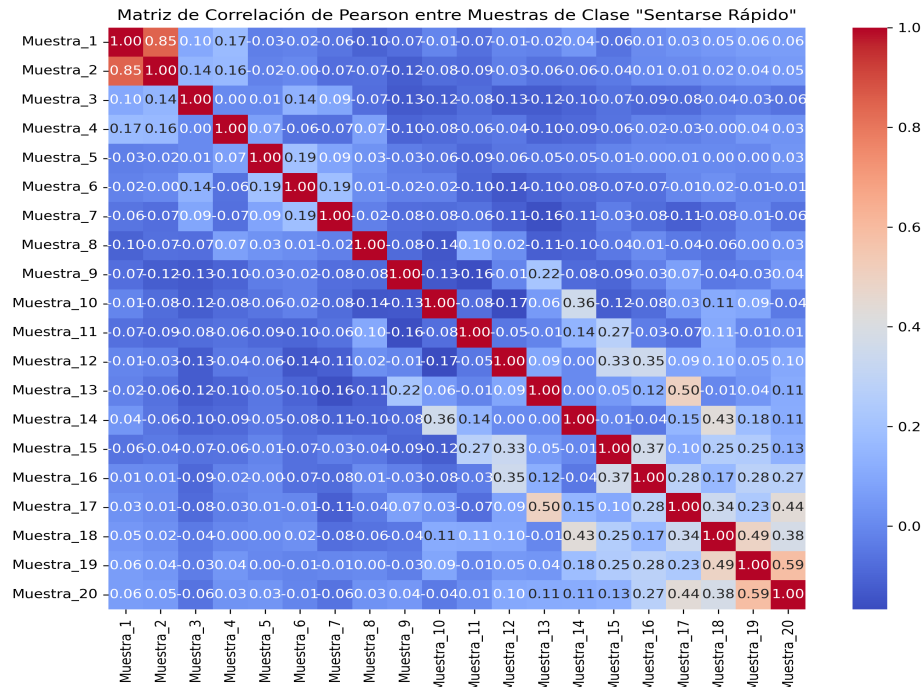


Figura 4.17: Gráfico de Indices de Correlación de Pearson para la clase Sentarse Rápido.

Como se puede observar en la figura anterior, la matriz de correlación de Pearson entre muestras de la clase Sentarse Rápido presenta un índice de correlación más bajo entre sus muestras, ya que durante la ejecución de movimientos como Caída, Agacharse, Caminar Rápido y Levantarse Rápido, es muy complicado que un gran grupo de muestras del movimiento sea similar. Esto se debe a que, al estar en la cinemática de un movimiento rápido, las aceleraciones y velocidades angulares cambian rápidamente. es por ello que podemos observar que solamente en las muestras 1 y 2 tenemos un índice de correlación de 0.85, mientras que en las demás muestras son, incluso, menores a 0.2 en su mayoría. Al final de la gráfica, podemos observar que, de la muestra 16 a la muestra 20, los índices de correlación aumentan otra vez hasta 0.59, debido a que en esas muestras está finalizando la ejecución del movimiento, por lo que se comienza a regresar a una posición de reposo, donde las muestras pueden empezar a acercarse a valores similares cuando la aceleración y la velocidad angular disminuyen.

El análisis detallado de las matrices de correlación resulta fundamental para comprender el grado de similitud y variabilidad presentes en las señales inerciales asociadas a cada clase de movimiento. Estos resultados no solo permiten identificar patrones consistentes

y diferenciables, sino que también proporcionan información clave para la optimización y validación del modelo de inferencia. En el contexto de un sistema neuromórfico, esta etapa es esencial, ya que garantiza que la red neuronal pueda reconocer con precisión las actividades realizadas por adultos mayores, incluso cuando existen variaciones naturales en la ejecución de los movimientos. Al contar con un modelo confiable y validado, se sientan las bases para el desarrollo de una herramienta de asistencia capaz de detectar y alertar sobre eventos críticos, como caídas, en tiempo real. Esto representa un paso significativo hacia la implementación de tecnologías que promuevan la seguridad, la autonomía y la calidad de vida de las personas mayores.

Buscando la optimización de recursos del microcontrolador en la inferencia del sistema neuromórfico, se creó un modelo adicional de Redes Neuronales para seleccionar únicamente los 50 datos más representativos de cada medición, a fin de utilizar un modelo más compacto.

4.4 Modelo de selección de características por XGBoost

Este método permite asignar un nivel de importancia a cada característica según su contribución a la precisión del modelo, favoreciendo la eliminación de datos redundantes o poco relevantes. De esta manera, se optimiza el rendimiento de la red neuronal al reducir la dimensionalidad de los datos de entrada, concentrando el aprendizaje en las señales con mayor peso discriminativo. A continuación, se presenta la tabla con las 50 características más importantes determinadas por el algoritmo XGBoost, las cuales constituyen el conjunto final de parámetros empleados para la fase de entrenamiento del sistema neuromórfico.

A continuación, en las tablas [Tabla 4.3](#) y [Tabla 4.4](#) se muestra la importancia de cada una de las características seleccionadas mediante el algoritmo de XGBoost, considerando que no están en orden cronológico, sino que se encuentran ordenadas de la característica seleccionada de la más importante a la menos importante dentro de las 50 estipuladas.

Tabla 4.3: Importancia de las 50 características más relevantes del modelo XGBoost

Feature	Importance
Ay52	0.088798
Ax74	0.054784
Az51	0.041669
Ax38	0.040171
Ay100	0.040067
Az57	0.032409
Ax18	0.032129
Az45	0.030890
Az47	0.027737
Ay65	0.027455
Ay40	0.022710
Ay50	0.021143
Ax65	0.014358
Ax96	0.012049
Vz37	0.011500
Vz74	0.011349
Ay4	0.010902
VR17	0.010414
Vx6	0.009709
Ax68	0.009205
Az73	0.008900
Ay30	0.008889
VR4	0.008847
Ay94	0.008761
Ay53	0.007823
Az63	0.007648

Tabla 4.4: Importancia de las 50 características más relevantes del modelo XGBoost

Feature	Importance
Az33	0.007552
Ax21	0.007374
Ay75	0.007287
Vy67	0.007216
VR19	0.006882
Ax3	0.006745
Ay41	0.006661
Ay67	0.006199
Az38	0.005858
Ay54	0.005573
Ay60	0.005298
Ax25	0.005155
Vz53	0.005080
Az100	0.005059
Ay83	0.004882
Az48	0.004818
Ay63	0.004696
Az59	0.004682
Ax52	0.004601
Vx62	0.004513
Ax41	0.004392
Az58	0.004385
VR57	0.004256
Vy59	0.004231

Como se puede observar en la [Tabla 4.3](#) y [Tabla 4.4](#), la característica más importante seleccionada por el algoritmo de XGBoost es la aceleración en el eje Y de la muestra 52 de cada una de las mediciones dentro de la base de datos, que, como ya se mencionó anteriormente, posee 50 mediciones por cada clase, siendo cada medición de 100 muestras originalmente, cada muestra con un bloque de 8 datos inerciales.

A continuación, en la [Tabla 4.5](#) se muestra la estructura de la red neuronal creada a partir de la selección de características más importantes por el algoritmo de XGBoost.

Tabla 4.5: Estructura final del modelo de red neuronal por XGBoost para 50 características más importantes.

Layer (type)	Output Shape	Param #
dense_16 (Dense)	(None, 256)	12,544
leaky_re_lu_12 (LeakyReLU)	(None, 256)	0
dropout_8 (Dropout)	(None, 256)	0
dense_17 (Dense)	(None, 128)	32,896
leaky_re_lu_13 (LeakyReLU)	(None, 128)	0
dropout_9 (Dropout)	(None, 128)	0
dense_18 (Dense)	(None, 64)	8,256
leaky_re_lu_14 (LeakyReLU)	(None, 64)	0
dense_19 (Dense)	(None, 11)	715

Total params: 54,411 Trainable params: 54,411 Non-trainable params: 0

Como se puede analizar en la Tabla 4.1, el modelo está conformado por 4 capas densas, siendo 11 categorías las que ingresan al modelo. La primera capa consiste en una capa de 256 neuronas y 40,704 parámetros, con una función de activación Leaky ReLU; posteriormente, se integra una capa densa de 128 neuronas y 40,704 parámetros. Después, se integra una capa densa de 64 neuronas y 8,256 parámetros, mientras que la última capa, es decir, la capa de salidas, contiene 11 neuronas y 715 parámetros.

De este modo, tenemos un modelo de Red Neuronal con 54,411 parámetros entrenables.

Tabla 4.6: Reporte de clasificación del modelo en la fase de validación

Clase	Precision	Recall	F1-score	Support
agacharse	1.00	1.00	1.00	10
caida_lateralD	1.00	1.00	1.00	10
caminar_lento	1.00	1.00	1.00	10
caminar_normal	0.82	0.90	0.86	10
caminar_rapido	0.90	0.90	0.90	10
levantarse_lento	1.00	1.00	1.00	10
levantarse_normal	0.92	0.86	0.89	14
levantarse_rapido	0.90	0.90	0.90	10
sentarse_lento	1.00	1.00	1.00	10
sentarse_normal	0.91	1.00	0.95	10
sentarse_rapido	0.89	0.80	0.84	10
Accuracy	0.94			114
Macro avg	0.94	0.94	0.94	114
Weighted avg	0.94	0.94	0.94	114

En la Tabla 4.6 se muestra el reporte de clasificación del modelo más reciente durante la fase de validación. Este modelo alcanzó una exactitud global del 94 %, con valores promedio de precisión, recall y F1-score de 0.94 tanto en el promedio macro como en el ponderado. Aunque la mayoría de las clases mantienen un rendimiento alto (con valores de F1-score iguales o cercanos a 1.00), se observa una disminución en el desempeño de las clases “caminar_normal” (F1=0.86), “levantarse_normal” (F1=0.89) y “sentarse_rapido” (F1=0.84). Estas reducciones indican que el modelo presenta cierta dificultad para discriminar entre actividades con patrones de movimiento similares o transiciones rápidas.

En contraste, el modelo anterior (Tabla 4.6) mostró un mejor desempeño general, alcanzando una exactitud del 96 % y valores promedio de F1-score entre 0.96 y 0.97. En dicho modelo, la mayoría de las clases fueron clasificadas correctamente con métricas perfectas (1.00), y aunque

algunas categorías como “levantarse_normal” presentaron un recall de 0.71, el rendimiento global fue superior.

Por lo tanto, se concluye que el modelo anterior ofrece un mejor desempeño en la fase de validación, mostrando una mayor precisión global y un balance más favorable entre precisión y sensibilidad. En consecuencia, este modelo fue seleccionado para su implementación en el sistema neuromórfico desarrollado, ya que garantiza una mayor confiabilidad en la clasificación de las actividades de los usuarios y una respuesta más consistente en condiciones reales de operación.

A continuación, se muestra en [Figura 4.18](#) un gráfico representativo del porcentaje de pérdida en el entrenamiento y la validación del modelo.



Figura 4.18: Gráfico representativo de pérdida en la fase de entrenamiento y fase de validación del modelo por XGBoost.

En la [Figura 4.18](#) se observa la evolución de la función de pérdida (loss) tanto en la fase de

entrenamiento como en la fase de validación del modelo basado en XGBoost. Al inicio del proceso (épocas 0 a 5) se aprecia una disminución pronunciada de ambas curvas, lo que indica que el modelo logra aprender rápidamente los patrones principales de los datos, reduciendo de forma significativa el error de predicción.

A partir de la época 10, las curvas de pérdida de entrenamiento y validación tienden a estabilizarse, mostrando valores cercanos entre sí, con ligeras oscilaciones en la validación. Este comportamiento sugiere que el modelo alcanzó un punto de convergencia adecuado, donde continúa mejorando gradualmente sin presentar signos evidentes de sobreajuste (overfitting), ya que la brecha entre ambas curvas se mantiene relativamente constante y pequeña.

La pérdida de entrenamiento alcanza valores ligeramente inferiores a los de validación, lo cual es un comportamiento esperado, ya que el modelo se ajusta de manera más precisa a los datos con los que fue entrenado. Sin embargo, la estabilidad observada en la curva de validación confirma que el modelo mantiene una buena capacidad de generalización sobre datos no vistos.

En resumen, el gráfico de pérdida refleja que el proceso de entrenamiento del modelo XGBoost fue eficiente y estable, logrando minimizar el error tanto en el conjunto de entrenamiento como en el de validación sin incurrir en una sobreoptimización del modelo.

Como continuación y complementación del desempeño del modelo de Red Neuronal, a continuación, en la [Figura 4.19](#) se muestra un gráfico de matriz de confusión del modelo evaluado en la fase de prueba o validación.

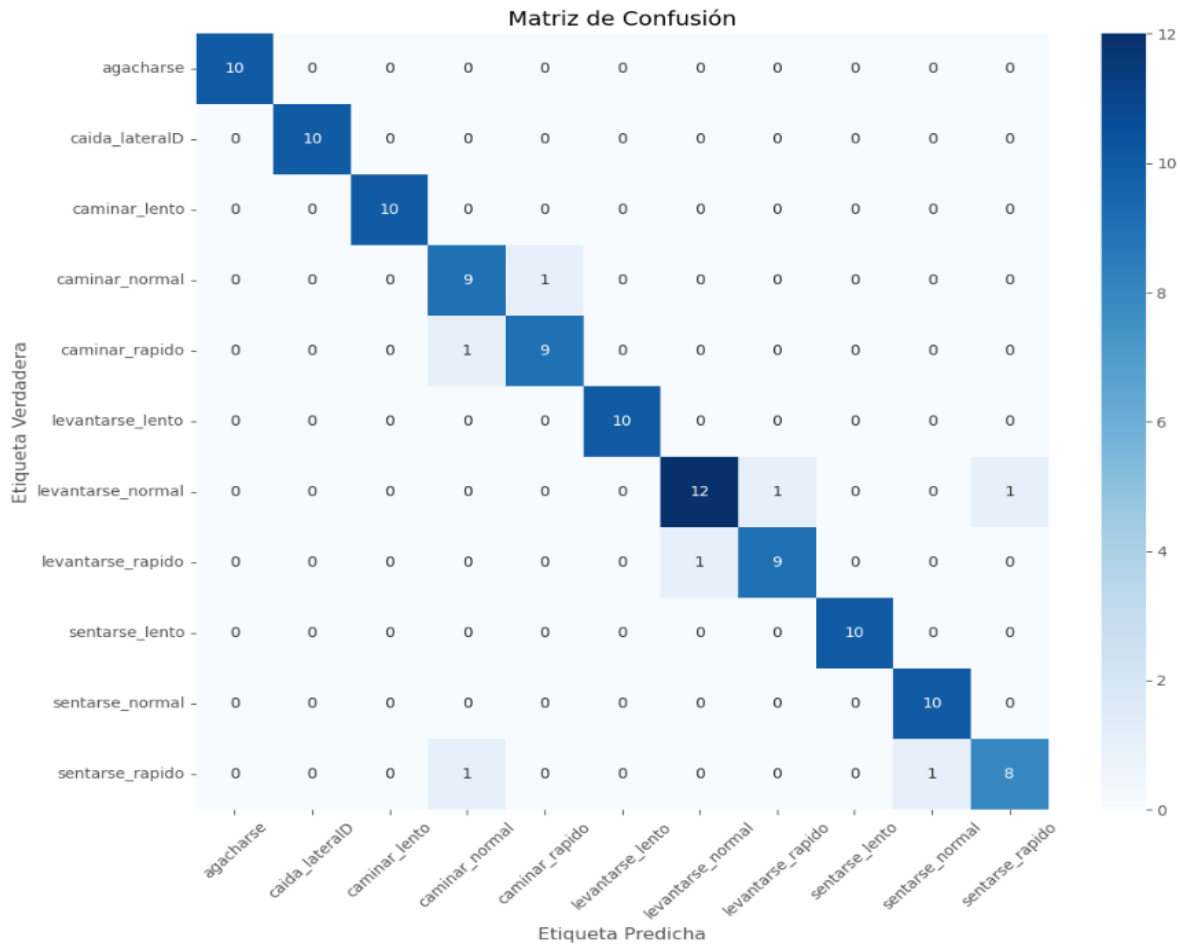


Figura 4.19: Matriz de confusión para validación del modelo de Red Neuronal de modelo XGBoost.

En la Figura 4.19 se presenta la matriz de confusión correspondiente al modelo durante la fase de validación. En ella se observa que la mayoría de las clases fue correctamente identificada, lo que se refleja en la diagonal principal, donde la mayoría de los valores se concentran con conteos de entre 9 y 12 aciertos por clase.

El modelo logra una clasificación perfecta en actividades como “agacharse”, “caida_lateralD”, “caminar_lento”, “levantarse_lento”, “sentarse_lento” y “sentarse_normal”, evidenciando su alta capacidad para reconocer patrones bien definidos y con diferencias cinemáticas claras.

Sin embargo, se aprecian algunas confusiones entre clases con características similares. Por ejemplo, la clase “caminar_normal” presenta una leve confusión con “caminar_rapido”, mien-

tras que “levantarse_normal” se confunde en una ocasión con “levantarse_rapido” y “sentarse_rapido”. Esta última también muestra errores en “caminar_normal” y “levantarse_normal”. Estas imprecisiones sugieren que el modelo tiene dificultades para distinguir entre transiciones rápidas o actividades con dinámicas temporales muy parecidas, donde los rangos de aceleración y desplazamiento son similares.

En general, la matriz de confusión confirma los resultados del reporte de clasificación: el modelo mantiene un desempeño elevado, con errores mínimos concentrados en clases que presentan una alta similitud en la forma y duración del movimiento. No obstante, estas confusiones explican la ligera disminución en la métrica global de precisión (94 %) respecto al modelo anterior (96 %), reafirmando que, aunque el rendimiento sigue siendo alto, el modelo previo mostró una capacidad de clasificación más precisa entre actividades cercanas.

En consecuencia, se determinó que el modelo anterior representa la opción más adecuada para su implementación en el sistema neuromórfico propuesto, al ofrecer un equilibrio óptimo entre precisión y estabilidad en la clasificación de actividades.

Para la implementación en el sistema neuromórfico en tiempo real, se seleccionó el modelo generado a partir del cálculo de energía por sumatoria del valor absoluto, en lugar del modelo derivado de la selección de características mediante XGBoost. Aunque el modelo basado en XGBoost produce un archivo .h más compacto, el enfoque por sumatoria del valor absoluto mostró un desempeño superior durante las fases de validación, reflejando un mayor porcentaje de precisión y una menor pérdida.

Esto ocurre porque, al aplicar la selección de características, se elimina gran parte del patrón cinemático de las aceleraciones y velocidades angulares. El algoritmo identifica características globales considerando las 11 clases en conjunto; sin embargo, dichas características son representativas solo de algunas actividades y no se generalizan bien hacia otras. Esto aumenta la correlación entre variables y reduce la capacidad del modelo para capturar diferencias reales entre clases durante la inferencia en tiempo real.

Además, este método presenta una estructura significativamente más sencilla de implementar en el Portenta H7, lo cual favorece su integración directa en el sistema embebido sin requerir

procesamiento adicional ni etapas complejas de prefiltrado. En conjunto, estos factores justifican la elección del modelo por sumatoria del valor absoluto como la opción más precisa y práctica para la operación en tiempo real del sistema neuromórfico.

4.5 Integración del sistema en herramienta de asistencia wearable

El sistema neuromórfico fue integrado en una hombrera concebida como una herramienta wearable de soporte destinada al monitoreo de actividades en adultos mayores. Esta hombrera incorpora un compartimento térmico, diseñado para una compresa caliente o fría, el cual fue aprovechado para alojar la unidad de alimentación y asegurar la protección física del hardware.

En su parte del compartimento se integró la unidad de alimentación del sistema, mientras que en la parte externa se adecuaron materiales para fijar el sistema embebido neuromórfico, que integra las unidades sensoriales (IMU) y la unidad de comunicación, distribuidos de manera que no comprometen la comodidad del usuario. La configuración final permite una operación autónoma y discreta del sistema, favoreciendo la detección y respuesta ante las actividades humanas.

A continuación, en la [Figura 4.20](#) se presenta la integración final del sistema neuromórfico como herramienta de asistencia wearable para adultos mayores.



Figura 4.20: Integración del sistema neuromórfico en la prenda vestible (wearable) con sus partes.

Como se puede observar en la [Figura 4.20](#), el sistema neuromórfico se logró integrar en la hombrera de forma adecuada para garantizar una comodidad para el usuario y mantener una ubicación segura para el sistema embebido neuromórfico a fin de mantener la posición de la Unidad de Medición Inercial a como se ubicó en la fase de registro y entrenamiento del modelo utilizado. Esta metodología nos garantiza que los datos a inferir estén normalizados en un estándar tanto de software como de hardware.

Complementariamente para evaluar el desempeño de la herramienta, a continuación se presenta el análisis de los tiempos de respuesta del proceso de inferencia y de la emisión de la alerta, con el fin de evaluar la eficiencia operativa de la herramienta en escenarios reales de uso.

4.6 Tiempo de respuesta del sistema

A continuación, se define el tiempo de respuesta del sistema neuromórfico como el intervalo comprendido entre la detección del evento de caída y la ejecución de la acción de alerta por parte del sistema. Este parámetro se evalúa desde dos perspectivas.

En primer lugar, se considera el tiempo de respuesta interno, que abarca desde la identificación

de la caída por la red neuromórfica hasta el momento en que el microcontrolador confirma el envío de la alerta. Esta medición permite analizar el rendimiento del procesamiento y la eficiencia de la comunicación del sistema embebido.

En segundo lugar, se define el tiempo de respuesta funcional, que comprende el intervalo total desde la ocurrencia física de la caída hasta la recepción efectiva de la notificación por parte del cuidador en su dispositivo móvil. Esta segunda métrica refleja el desempeño real del sistema en un entorno de aplicación, ya que incluye los retardos asociados a la red de comunicación y a la entrega de mensajes.

En las pruebas experimentales se empleará principalmente el tiempo de respuesta interno, por ser un parámetro controlable y reproducible dentro del entorno de desarrollo, mientras que el tiempo de respuesta funcional se considerará como una medida complementaria para la validación en condiciones reales.

A continuación, en la [Tabla 4.7](#) se muestra una tabla del promedio de tiempos de inferencia y envío de alertas del sistema neuromórfico desarrollado.

Tabla 4.7: Resultados de tiempo de inferencia y envío del sistema neuromórfico

Medición	Inferencia y envío (ms)
M1	6.987
M2	7.729
M3	6.791
M4	6.519
M5	6.811
PROMEDIO	6.9674

En la [Tabla 4.7](#) se presentan los tiempos medidos, en milisegundos, correspondientes al proceso completo de inferencia del modelo neuromórfico y al envío del comando AT al módulo SIM800L para notificar la predicción obtenida. Los resultados muestran una consistencia temporal adecuada entre las diferentes mediciones, con valores que oscilan entre 6.519 ms y 7.729 ms, y un promedio general de 6.9674 ms.

Este tiempo incluye tanto la ejecución del modelo de inferencia en el microcontrolador Portenta H7 Lite como la instrucción de transmisión inicial al módulo GSM. La baja dispersión entre mediciones indica una estabilidad en el desempeño del sistema, evidenciando que el proceso de inferencia y comunicación se realiza de manera eficiente y sin demoras significativas.

El tiempo promedio obtenido demuestra que el sistema es capaz de realizar el reconocimiento de actividad y preparar la notificación en un intervalo inferior a 10 milisegundos, lo cual resulta altamente favorable para aplicaciones de detección de caídas en tiempo real. Este desempeño permite que la respuesta global del sistema dependa principalmente de los tiempos de transmisión del módulo SIM800L y no del procesamiento interno, garantizando así una rápida reacción ante eventos críticos.

A continuación, en la [Tabla 4.8](#) se presentan los tiempos de respuesta promedio para el envío de alertas por SMS y por llamada telefónica a través del módulo de comunicación GSM.

Tabla 4.8: Tiempos de respuesta del sistema neuromórfico para envío de alertas

Medición	Alerta SMS (s)	Alerta Llamada (s)
M1	9.347	14.457
M2	8.875	13.962
M3	8.635	13.345
M4	8.134	14.245
M5	8.156	13.765
PROMEDIO	8.6294	13.9548

En la [Tabla 4.8](#) se presentan los tiempos de respuesta medidos para el envío de alertas mediante SMS y llamada telefónica, utilizando el módulo SIM800L.

El promedio de tiempo de respuesta para la alerta por SMS fue de 8.6294 segundos, mientras que para la llamada fue de 13.9548 segundos.

Estos valores reflejan el intervalo total transcurrido desde el momento en que el sistema neuromórfico genera la instrucción de notificación (posterior a la inferencia) hasta que la alerta

es recibida en el dispositivo del cuidador. A diferencia del tiempo de inferencia, que ocurre en el orden de milisegundos, estos tiempos dependen principalmente de la latencia de la red GSM, el proceso de registro en la red y la transmisión de datos propia del servicio móvil.

Aunque las alertas presentan un retardo de varios segundos, este comportamiento es aceptable para aplicaciones de monitoreo en tiempo real no crítico, como la detección y notificación de caídas en adultos mayores, donde el objetivo principal es la emisión oportuna de una alerta, no el control inmediato de un proceso físico.

Por tanto, puede afirmarse que el sistema cumple con los requisitos de respuesta en tiempo real a nivel de procesamiento y ofrece un tiempo de notificación adecuado, considerando las limitaciones de comunicación inherentes al módulo GSM.

4.7 Consumo y eficiencia energética del sistema

A continuación, en la [Tabla 4.9](#) se presenta el consumo estimado de corriente de cada componente del sistema neuromórfico con el análisis propuesto en la Etapa 6 del Capítulo 3 del presente trabajo, correspondiente a la fase de pruebas y validación del sistema desarrollado.

Tabla 4.9: Consumo estimado de corriente de cada componente del sistema neuromórfico

Componente	Consumo típico (mA)	Escenario	Notas
Portenta H7 Lite	80	Activo (inferencia)	Corriente estimada durante procesamiento normal
MPU6050	4	Operación continua	Sensor en modo activo
SIM800L	10	En espera (idle)	Registrado en red sin transmisión
SIM800L	500	Transmisión promedio	Envío de SMS o datos
SIM800L	800	Pico durante llamada	Corriente típica durante comunicación por voz
SIM800L	2000	Pico transitorio	Breve ráfaga durante transmisión GSM
PAM8403	10	Reposo (sin audio)	Corriente quiescente
PAM8403	150	Reproducción de audio	Volumen medio con bocina de 8 Ω

El análisis de la [Tabla 4.9](#) permite observar que el consumo energético del sistema neuromórfico está dominado principalmente por los picos de corriente del módulo SIM800L durante los eventos de transmisión y llamada, alcanzando valores transitorios de hasta 2000 mA, mientras que el resto de los componentes presentan consumos considerablemente menores. En condiciones de operación normal, el Portenta H7 Lite y el módulo de audio PAM8403 en reproducción constituyen las fuentes de consumo más relevantes después del módulo GSM. Estos valores resultan fundamentales para estimar la autonomía del sistema, ya que permiten establecer los escenarios de demanda energética en reposo y durante la ejecución de tareas críticas, análisis que se desarrolla en la siguiente sección mediante el cálculo de la duración promedio de la batería. (Ver [Tabla 4.10](#))

Tabla 4.10: Estimación de autonomía de la batería de 2500 mAh (3.7 V) en distintos escenarios

Escenario	Corriente total (mA)	Duración estimada (h)
Monitoreo (reposo / bajo consumo)	104	24.0
Operación activa (inferencia + SMS + audio)	734	3.4
Transmisión GSM (llamada / ráfagas altas)	1034	2.4
Caso extremo (picos continuos de 2 A)	2234	1.1

De acuerdo con la [Tabla 4.10](#), se aprecia que la autonomía del sistema neuromórfico varía significativamente en función del escenario operativo. En condiciones de monitoreo o reposo, donde el sistema mantiene únicamente los módulos esenciales activos, la batería de 2500 mAh puede alcanzar una duración aproximada de 24 horas continuas. Sin embargo, durante la operación activa que incluye la inferencia del modelo, el envío de alertas por SMS y la reproducción de audio la autonomía se reduce a poco más de tres horas. En escenarios de transmisión GSM o llamadas, el consumo se incrementa notablemente debido a las ráfagas de corriente del módulo SIM800L, limitando la duración a aproximadamente dos horas y media. Finalmente, bajo condiciones extremas de picos sostenidos de corriente, la batería apenas alcanzaría poco más de una hora de operación, lo que evidencia la necesidad de una gestión eficiente de la energía para mantener la funcionalidad del sistema durante periodos prolongados.

En conjunto, los resultados obtenidos permiten consolidar una visión integral del desempeño del sistema propuesto. La comparación entre el análisis del patrón cinemático de una caída y las salidas generadas por ambos modelos de red neuronal evidencia una coherencia sólida entre la dinámica real del evento y las predicciones del modelo con mayor precisión y facilidad de implementación. Asimismo, el estudio de las matrices de correlación permitió identificar las variables con mayor peso en la discriminación de actividades, fortaleciendo la interpretación de los resultados y la justificación del modelo seleccionado. Finalmente, el análisis de tiempos

de respuesta y consumo energético del sistema embebido confirmó la viabilidad operativa del diseño, demostrando que la inferencia puede ejecutarse en tiempo real sin comprometer la autonomía energética. Estos hallazgos consolidan la eficacia del sistema y sientan las bases para su optimización e integración en aplicaciones de monitoreo continuo.

CONCLUSIONES

El desarrollo del sistema neuromórfico propuesto permitió integrar técnicas de inteligencia artificial embebida para el reconocimiento de actividades humanas, con énfasis en la detección de caídas en adultos mayores. El análisis inercial, tanto en su etapa inicial como postprocesada, resultó fundamental para la correcta caracterización de los patrones cinemáticos asociados a cada movimiento, facilitando un entrenamiento representativo y estable del modelo.

La metodología utilizada para el desarrollo de nuestro trabajo permitió obtener un sistema embebido con la capacidad de inferir diferentes actividades humanas propias de los adultos mayores, tales como las que se han descrito anteriormente.

Los resultados obtenidos confirman que el modelo Multilayer Perceptron (MLP) demostró un desempeño superior en precisión y capacidad de predicción o inferencia de las clases consideradas (actividades humanas) en comparación con el método basado en la selección de características mediante XGBoost. Adicionalmente, se confirmó que hay una menor complejidad en la implementación en plataformas con recursos limitados de hardware.

La selección de las categorías o actividades humanas en adultos mayores que se consideraron en este trabajo se realizó con base a estudios formales dentro del campo geriátrico, estableciendo un total de once categorías.

En este sentido, los resultados alcanzados demuestran que, para la categoría crítica de caída (adulto mayor), se obtiene la más alta precisión (100 %)

Asimismo, los resultados del tiempo de inferencia y de respuesta a las alertas validaron la viabilidad del sistema para operar en tiempo real dentro de un dispositivo portátil, manteniendo un equilibrio adecuado entre el rendimiento computacional y la autonomía operativa.

En conjunto, este trabajo demuestra la factibilidad de implementar un sistema neuromórfico embebido para el monitoreo de actividades humanas, aportando una solución práctica para la asistencia y detección temprana de eventos críticos en adultos mayores.

Finalmente, los resultados obtenidos abren la puerta a trabajos futuros orientados a la optimización del hardware, integración con redes 4G o Wi-Fi, técnicas de entrenamiento incremental y

la miniaturización del dispositivo, con el fin de mejorar su confort, autonomía y escalabilidad para su uso cotidiano.

REFERENCIAS BIBLIOGRÁFICAS

- [1] B. de Publicaciones Oficiales del Gobierno de la República, *Envejecimiento y salud. Una propuesta para un plan de acción*, mayo de 2018.
- [2] C. N. de Población, *Día Internacional de las Personas de Edad*, 2021.
- [3] O. M. de Salud, *Caídas*, abr. de 2021.
- [4] C. C. Lim et al., «Smart fall detection monitoring system using wearable sensor and Raspberry Pi,» en *AIP Conference Proceedings*, vol. 2898, 2024. DOI: [10.1063/5.0192471](https://doi.org/10.1063/5.0192471).
- [5] I. Goldwin, A. Gunadhi, D. Lestariningsih, H. Pranjoto, P. Rhatodirdjo y L. Agustine, «Fall detection and notification system to fast emergency management for the elderly,» en *E3S Web of Conferences*, vol. 475, 2024. DOI: [10.1051/e3sconf/202447502014](https://doi.org/10.1051/e3sconf/202447502014).
- [6] G. Ascioğlu e Y. Senol, «Design of a wearable wireless multi-sensor monitoring system and application for activity recognition using deep learning,» *IEEE Access*, vol. 8, 2020, ISSN: 21693536. DOI: [10.1109/ACCESS.2020.3024003](https://doi.org/10.1109/ACCESS.2020.3024003).
- [7] B. Coffen y M. S. Mahmud, «TinyDL: Edge computing and deep learning based real-time hand gesture recognition using wearable sensor,» en *2020 IEEE International Conference on E-Health Networking, Application and Services, HEALTHCOM 2020*, 2021. DOI: [10.1109/HEALTHCOM49281.2021.9399005](https://doi.org/10.1109/HEALTHCOM49281.2021.9399005).
- [8] S. Perez-Gamboa, Q. Sun e Y. Zhang, «Improved Sensor Based Human Activity Recognition via Hybrid Convolutional and Recurrent Neural Networks,» en *INERTIAL 2021 - 8th IEEE International Symposium on Inertial Sensors and Systems, Proceedings*, 2021. DOI: [10.1109/INERTIAL51137.2021.9430460](https://doi.org/10.1109/INERTIAL51137.2021.9430460).
- [9] A. Shakerian, V. Douet, A. S. Nejati y R. Landry, «Real-Time Sensor-Embedded Neural Network for Human Activity Recognition,» *Sensors*, vol. 23, 19 2023, ISSN: 14248220. DOI: [10.3390/s23198127](https://doi.org/10.3390/s23198127).

- [10] T. H. Tan, J. Y. Shih, S. H. Liu, M. Alkhaleefah, Y. L. Chang y M. Gochoo, «Using a Hybrid Neural Network and a Regularized Extreme Learning Machine for Human Activity Recognition with Smartphone and Smartwatch,» *Sensors*, vol. 23, 6 2023, ISSN: 14248220. DOI: [10.3390/s23063354](https://doi.org/10.3390/s23063354).
- [11] S. Wan, L. Qi, X. Xu, C. Tong y Z. Gu, «Deep Learning Models for Real-time Human Activity Recognition with Smartphones,» *Mobile Networks and Applications*, vol. 25, 2 2020, ISSN: 15728153. DOI: [10.1007/s11036-019-01445-x](https://doi.org/10.1007/s11036-019-01445-x).
- [12] A. Prasad, A. K. Tyagi, M. M. Althobaiti, A. Almulihi, R. F. Mansour y A. M. Mahmoud, «Human activity recognition using cell phone-based accelerometer and convolutional neural network,» *Applied Sciences (Switzerland)*, vol. 11, 24 2021, ISSN: 20763417. DOI: [10.3390/app112412099](https://doi.org/10.3390/app112412099).
- [13] «Recurrent neural network for human activity recognition in embedded systems using ppg and accelerometer data,» *Electronics (Switzerland)*, vol. 10, 14 2021, ISSN: 20799292. DOI: [10.3390/electronics10141715](https://doi.org/10.3390/electronics10141715).
- [14] M. Bennasar, B. A. Price, D. Gooch, A. K. Bandara y B. Nuseibeh, «Significant Features for Human Activity Recognition Using Tri-Axial Accelerometers,» *Sensors*, vol. 22, 19 2022, ISSN: 14248220. DOI: [10.3390/s22197482](https://doi.org/10.3390/s22197482).
- [15] S. Zhang et al., «Deep Learning in Human Activity Recognition with Wearable Sensors: A Review on Advances,» *Sensors*, vol. 22, 4 2022, ISSN: 14248220. DOI: [10.3390/s22041476](https://doi.org/10.3390/s22041476).
- [16] S. Ward, S. Hu y M. Zecca, «Effect of Equipment on the Accuracy of Accelerometer-Based Human Activity Recognition in Extreme Environments,» *Sensors*, vol. 23, 3 2023, ISSN: 14248220. DOI: [10.3390/s23031416](https://doi.org/10.3390/s23031416).
- [17] Y. J. Luwe, C. P. Lee y K. M. Lim, «Wearable Sensor-Based Human Activity Recognition with Hybrid Deep Learning Model,» *Informatics*, vol. 9, 3 2022, ISSN: 22279709. DOI: [10.3390/informatics9030056](https://doi.org/10.3390/informatics9030056).
- [18] K. P. Seng, L. M. Ang, E. Peter y A. Mmonyi, *Machine Learning and AI Technologies for Smart Wearables*, 2023. DOI: [10.3390/electronics12071509](https://doi.org/10.3390/electronics12071509).

- [19] OMS, *Envejecimiento y Salud*, oct. de 2024.
- [20] M. Halpin, *Assistive Technology for the Elderly: Devices to Improve Daily Life*, oct. de 2023. dirección: <https://reciteme-com.translate.google/news/assistive-technology-for-the-elderly>.
- [21] G. Galeano, *Programación de sistemas embebido en C*, 1.ª ed. Alfaomega Grupo Editor, jun. de 2009.
- [22] J. Dunn, R. Runge y M. Snyder, *Wearables and the medical revolution*, 2018. DOI: [10.2217/pme-2018-0044](https://doi.org/10.2217/pme-2018-0044).
- [23] N. Niknejad, W. B. Ismail, A. Mardani, H. Liao e I. Ghani, «A comprehensive overview of smart wearables: The state of the art literature, recent advances, and future challenges,» *Engineering Applications of Artificial Intelligence*, vol. 90, 2020, ISSN: 09521976. DOI: [10.1016/j.engappai.2020.103529](https://doi.org/10.1016/j.engappai.2020.103529).
- [24] F. Alós y A. Puig-Ribera, «Clinical use of wearables and Mobile Apps (mHealth) to change patient's lifestyles through a primary care-based approach: A narrative review,» *Elsevier*, 2021.
- [25] CanalSenior, *Wearables para mejorar la salud en personas mayores: una guía completa*, dic. de 2023. dirección: <https://canalsenior.es/noticias/wearebles-personas-mayores-mejorar-salud>.
- [26] C. Stryker y E. Kavlakoglu, *¿Qué es la inteligencia artificial o IA?* Ago. de 2024.
- [27] Scikit-learn, *Modelos de redes neuronales (supervisadas)*, 2024.
- [28] A. Ghibellini, L. Bononi y M. D. Felice, «Intelligence at the IoT Edge: Activity Recognition with Low-Power Microcontrollers and Convolutional Neural Networks,» en *Proceedings - IEEE Consumer Communications and Networking Conference, CCNC*, 2022. DOI: [10.1109/CCNC49033.2022.9700665](https://doi.org/10.1109/CCNC49033.2022.9700665).
- [29] IBM, *¿Qué son las redes neuronales?* Ago. de 2023.
- [30] I. Goodfellow, Y. Bengio y A. Courville, *Deep Learning*. MIT Press, 2016, <http://www.deeplearningbook.org>.

- [31] F. Cleary y Microchip, «Capacitive touch sensor design guide,» *Microchip AN2934*, online available at: <http://ww1.microchip.com/downloads/en/DeviceDoc/AN2934-Capacitive-Touch-Sensor-Design-Guide-00002934A.pdf>, 2020.
- [32] T. Chen y C. Guestrin, «XGBoost: A scalable tree boosting system,» en *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vol. 13-17-August-2016, 2016. DOI: [10.1145/2939672.2939785](https://doi.org/10.1145/2939672.2939785).

ANEXOS

4.8 ANEXO A

Comandos AT

Para realizar la comunicación, en este caso, con un microcontrolador, es necesario introducir en la parte inferior del módulo una micro SIM de alguna compañía telefónica que soporte la comunicación GSM.

El lenguaje en el que estos dispositivos se comunican es un lenguaje desarrollado por la compañía Hayes Communications que prácticamente se convirtió en un estándar abierto de comandos para configurar y parametrizar módems.

Los caracteres “AT”, que preceden a todos los comandos, significan “Atención”, e hicieron que se conociera también a este conjunto de comandos como comandos AT.

Algunos de los comandos AT para utilizar y configurar el módulo son los siguientes:

AT

Este es el comando base de los comandos Hayes. Con él, comprobamos la disponibilidad del dispositivo; es una manera de comprobar que el módulo está recibiendo señal y que se encuentra funcionando correctamente.

ATD

Este comando se usa típicamente para realizar una llamada.

ATH

Este comando se usa típicamente para colgar la llamada en curso.

AT+CMGS

Nos permite indicar el número de teléfono al que se va a enviar el SMS o mensaje de texto.

AT+CMGF=1

Nos permite configurar el módulo GSM como en modo de envío de mensajes de texto (SMS).

AT+CSQ

Este comando sirve para conocer la calidad del nivel de señal recibida por el dispositivo.

El nivel de la señal en unidades de Decibels se dan a partir de la [Ecuación 4.1](#):

$$\text{Nivel de señal (dBm)} = (\text{Nivel CSQ} \times 2) - 113 \quad (4.1)$$

A continuación en la [Tabla 4.11](#) se muestran dichas equivalencias de señal en dBm.

Tabla 4.11: Niveles de señal (CSQ) y su equivalencia en dBm del módulo SIM800L

Valor CSQ	Nivel de señal (dBm)	Descripción / Calidad de señal
0	-113 o menor	Sin señal
1	-111	Señal extremadamente débil
2	-109	Muy mala
3	-107	Muy mala
4	-105	Muy mala
5	-103	Muy mala
6	-101	Muy mala
7	-99	Muy mala
8	-97	Mala
9	-95	Mala
10	-93	Aceptable
11	-91	Aceptable
12	-89	Aceptable
13	-87	Buena
14	-85	Buena
15	-83	Buena
16	-81	Buena
17	-79	Buena
18	-77	Muy buena
19	-75	Muy buena
20	-73	Muy buena
21	-71	Excelente
22	-69	Excelente
23	-67	Excelente
24	-65	Excelente
25	-63	Excelente
30	-53 o mayor	Señal máxima
99	N/A	No detectable o error

4.9 ANEXO B

Algoritmo para selección de características más representativas mediante calculo de energía por sumatoria de valor absoluto

```
1 import pandas as pd
2 import numpy as np
3
4 # Parámetros
5 input_file = r'C:\Users\Juan Pablo\Desktop\BaseDatos800.csv'
6 output_file = r'C:\Users\Juan Pablo\Desktop\BaseDatos160.csv'
7
8 num_bloques_deseados = 20 # debe ser múltiplo de 8 bloques, ejemplo:
   60 bloques de 8 columnas = 480 columnas
9 tam_bloque = 8
10
11 # Leer el archivo CSV (sin modificar separadores, asume ',' por
   defecto)
12 df = pd.read_csv(input_file, header=0)
13
14 # Separar etiquetas y datos
15 etiquetas = df.iloc[:, -1].values # última columna
   etiquetas
16 datos_raw = df.iloc[:, :-1].astype(str).apply(lambda x: x.str.strip())
   # limpiar espacios
17
18 # Convertir a float
19 datos = datos_raw.apply(pd.to_numeric, errors='raise').values
20
21 total_columnas = datos.shape[1]
22 num_bloques = total_columnas // tam_bloque
23
24
25
26 assert total_columnas % tam_bloque == 0, f"Total columnas ({
   total_columnas}) no es múltiplo de {tam_bloque}"
```

```

27 assert num_bloques_deseados <= num_bloques, f"num_bloques_deseados ({
    num_bloques_deseados}) no puede ser mayor que {num_bloques}"
28
29 # Procesar fila por fila
30 datos_reducidos = []
31 for fila in datos:
32     bloques = np.split(fila, num_bloques) # dividir en bloques de 8
33     # Calcular "importancia" por bloque, usando suma de valor absoluto
34     # (puedes cambiar esta métrica)
35     pesos = np.array([np.sum(np.abs(b)) for b in bloques])
36
37     # Obtener índices de bloques más significativos
38     idx_top = np.argsort(pesos)[-num_bloques_deseados:]
39     idx_top.sort() # ordenar para mantener el orden temporal
40
41     # Seleccionar bloques
42     bloques_seleccionados = [bloques[i] for i in idx_top]
43
44     # Concatenar bloques seleccionados para formar la fila reducida
45     fila_reducida = np.concatenate(bloques_seleccionados)
46     datos_reducidos.append(fila_reducida)
47
48 # Crear DataFrame resultado
49 df_reducido = pd.DataFrame(datos_reducidos)
50
51 # Añadir la columna de etiquetas
52 df_reducido['Etiqueta'] = etiquetas[:df_reducido.shape[0]]
53
54 # Guardar resultado
55 df_reducido.to_csv(output_file, index=False)
56
57 print(f"Archivo reducido guardado como '{output_file}' con {
    df_reducido.shape[0]} filas y {df_reducido.shape[1]} columnas (
    incluye etiqueta).")

```

Listing 4.1: Algoritmo para selección de características más representativas mediante calculo de energía por sumatoria de valor absoluto

4.10 ANEXO C

Algoritmo para Red Neuronal MLP por cálculo de energía mediante sumatoria de valor absoluto

```
1 import pandas as pd
2 import numpy as np
3 from sklearn.model_selection import train_test_split
4 from sklearn.preprocessing import StandardScaler, LabelEncoder
5 from tensorflow.keras.models import Sequential
6 from tensorflow.keras.layers import Dense, Dropout
7 from tensorflow.keras.optimizers import Adam
8 from tensorflow.keras.utils import to_categorical
9 from tensorflow.keras.regularizers import l2
10 import joblib
11
12 # Cargar los datos
13 df = pd.read_csv('BaseDatos_160.csv')
14 df.head()
15
16 df.values[:,1:-1]
17
18 # # Separar características y etiqueta
19 df2 = df.drop(["MOV_OUT"], axis=1)
20 X=df2.values[:,1:-1]
21 y = df['MOV_OUT'].values # Última columna
22
23 print(X)
24
25 # Preprocesamiento
26 # Codificar etiquetas
```

```

27 le = LabelEncoder()
28 y_encoded = le.fit_transform(y)           # Etiquetas en formato
      entero
29 y_categorical = to_categorical(y_encoded)  # Conversión a one-hot para
      el modelo
30
31 # Verificar que tenemos 11 categorías
32 num_classes = len(le.classes_)
33 if num_classes != 11:
34     print(f"Advertencia: Se esperaban 11 categorías pero se
      encontraron {num_classes}")
35
36 # Dividir datos con estratificación
37 X_train, X_test, y_train, y_test = train_test_split(
38     X,
39     y_categorical,          # Se entrena con etiquetas one-hot
40     test_size=0.2,
41     random_state=42,
42     stratify=y_encoded     # Estratificación usando etiquetas enteras
43 )
44
45
46 print(X_train.shape)
47 print(y_train.shape)
48
49 for i, clase in enumerate(le.classes_):
50     print(f"{i}: {clase}")
51
52
53 # Normalizar características
54 scaler = StandardScaler()
55 X_train = scaler.fit_transform(X_train)
56 X_test = scaler.transform(X_test)
57
58 # Crear el modelo de red neuronal
59 from tensorflow.keras.layers import LeakyReLU

```



```

60
61 model = Sequential([
62     Dense(256, kernel_regularizer=l2(0.001), input_shape=(158,)),
63     LeakyReLU(alpha=0.01),
64     Dropout(0.1),
65     Dense(128),
66     LeakyReLU(alpha=0.01),
67     Dropout(0.1),
68     Dense(64, kernel_regularizer=l2(0.001)),
69     LeakyReLU(alpha=0.01),
70     Dense(num_classes, activation='softmax')
71 ])
72
73 # Compilar el modelo
74 model.compile(optimizer=Adam(learning_rate=0.001),
75               loss='categorical_crossentropy',
76               metrics=['accuracy'])
77
78 from tensorflow.keras.callbacks import EarlyStopping
79
80 early_stop = EarlyStopping(monitor='val_loss', patience=10,
81                             restore_best_weights=True)
82
83 history = model.fit(X_train, y_train,
84                     epochs=100,
85                     batch_size=32,
86                     validation_split=0.1,
87                     callbacks=[early_stop],
88                     verbose=1)
89
90 # Evaluar el modelo
91 test_loss, test_acc = model.evaluate(X_test, y_test, verbose=0)
92 print(f'\nPrecisión en conjunto de prueba: {test_acc:.4f}')
93
94 # Guardar el modelo entrenado (opcional)
95 model.save('movement_classifier.h5')

```

```

95
96 # 1. GRÁFICO DE LA FUNCIÓN DE PÉRDIDA DURANTE EL ENTRENAMIENTO
97 import matplotlib.pyplot as plt
98 import seaborn as sns
99
100 plt.figure(figsize=(12, 6))
101 plt.plot(history.history['loss'], label='Training Loss')
102 plt.plot(history.history['val_loss'], label='Validation Loss')
103 plt.title('Training and Validation Loss')
104 plt.xlabel('Epoch')
105 plt.ylabel('Loss')
106 plt.legend()
107 plt.grid(True)
108 plt.show()
109
110 # EVALUAR MODELO
111
112 test_loss, test_acc = model.evaluate(X_test, y_test, verbose=0)
113 print(f'\nPrecisión en conjunto de prueba: {test_acc:.4f}')
114 print(f'\nPérdida en conjunto de prueba: {test_loss:.4f}')
115
116 # PREDECIR CLASES PARA EL CONJUNTO DE PRUEBA
117
118 y_pred = model.predict(X_test)
119 y_pred_classes = np.argmax(y_pred, axis=1)
120 y_true_classes = np.argmax(y_test, axis=1)
121
122 # MATRIZ DE CONFUSIÓN
123 from sklearn.metrics import confusion_matrix, classification_report,
    f1_score
124 from tensorflow.keras.utils import to_categorical
125
126 conf_matrix = confusion_matrix(y_true_classes, y_pred_classes)
127 plt.figure(figsize=(12, 10))
128 sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
129 xticklabels=le.classes_, yticklabels=le.classes_)

```

```

130 plt.title('Matriz de Confusión')
131 plt.ylabel('Etiqueta Verdadera')
132 plt.xlabel('Etiqueta Predicha')
133 plt.xticks(rotation=45)
134 plt.yticks(rotation=0)
135 plt.show()
136
137 # MÉTRICAS DE CLASIFICACIÓN Y F1 SCORE
138
139 print("\nReporte de Clasificación:")
140 print(classification_report(y_true_classes, y_pred_classes,
141                             target_names=le.classes_))
142
143 # CALCULAR EL F1 SCORE MACRO Y WEIGHTED
144
145 f1_macro = f1_score(y_true_classes, y_pred_classes, average='macro')
146 f1_weighted = f1_score(y_true_classes, y_pred_classes, average='
147                     weighted')
148
149 print(f"\nF1-score Macro: {f1_macro:.4f}")
150 print(f"F1-score Weighted: {f1_weighted:.4f}")
151
152 # GUARDAR EL MODELO ENTRENADO Y SCALER
153
154 model.save('movement_classifier.h5')
155 joblib.dump scaler, 'scaler.pkl')
156
157 # INFORMACIÓN ADICIONAL SOBRE EL MODELO
158
159 print("\nResumen del modelo:")
160 model.summary()
161
162 # -----
163
164 from sklearn.manifold import TSNE
165 import matplotlib.pyplot as plt
166 import seaborn as sns
167 from sklearn.cluster import KMeans

```

```

164 # Reducir dimensionalidad con t-SNE
165 tsne = TSNE(n_components=2, random_state=42, perplexity=30, n_iter
    =1000)
166 X_tsne = tsne.fit_transform(X_test)
167
168 # Etiquetas reales (convertimos números a nombres con LabelEncoder)
169 y_true_classes = np.argmax(y_test, axis=1)
170 y_true_labels = le.inverse_transform(y_true_classes)
171
172 plt.figure(figsize=(12, 6))
173 sns.scatterplot(x=X_tsne[:,0], y=X_tsne[:,1],
174                hue=y_true_labels, palette="tab10", s=60)
175 plt.title("Etiquetas reales en 2D (t-SNE)")
176 plt.xlabel("Dim 1")
177 plt.ylabel("Dim 2")
178 plt.legend(title="Clases", bbox_to_anchor=(1.05, 1), loc='upper left')
179 plt.show()
180
181 # KMeans con mismo número de clusters que clases
182 kmeans = KMeans(n_clusters=num_classes, random_state=42)
183 clusters = kmeans.fit_predict(X_test)
184
185 # Gráfico de clusters KMeans
186 plt.figure(figsize=(12, 6))
187 sns.scatterplot(x=X_tsne[:,0], y=X_tsne[:,1],
188                hue=clusters, palette="tab10", s=60)
189 plt.title("Clusters de KMeans en 2D (t-SNE)")
190 plt.xlabel("Dim 1")
191 plt.ylabel("Dim 2")
192 plt.legend(title="Clusters", bbox_to_anchor=(1.05, 1), loc='upper left
    ')
193 plt.show()

```

Listing 4.2: Algoritmo para Red Neuronal Feedforward por cálculo de energía mediante sumatoria de valor absoluto

4.11 ANEXO D

Algoritmo para Red Neuronal Feedforward por selección de características mediante XGBoost

```
1
2 import numpy as np
3 import pandas as pd
4 import matplotlib.pyplot as plt
5 import seaborn as sns
6 from sklearn.model_selection import train_test_split
7 from sklearn.ensemble import RandomForestClassifier
8 from sklearn.multioutput import MultiOutputClassifier
9 from sklearn.feature_selection import SelectKBest, mutual_info_classif
10 from sklearn.decomposition import PCA
11 from sklearn.preprocessing import StandardScaler
12 from sklearn.metrics import accuracy_score
13 import xgboost as xgb
14 from xgboost import plot_importance
15 import shap
16 import time
17
18 import warnings
19 warnings.filterwarnings('ignore')
20
21 NumDatos=50; #Escribir el número de columnas a analizar de la DB
22
23
24 # Leer y preparar los datos (igual que en tu código original)
25 DataF1 = pd.read_csv("DataBase/BaseDatos.csv")
26
27 # Reducción de columnas Seleccionar 1 de cada 4 columnas usando iloc
28 ColDat=int(np.round(DataF1.shape[1]/NumDatos))
29 print(ColDat)
30 df_reducido = DataF1.iloc[:, ::ColDat] # Todos los renglones, 1
    columna cada 4
```

```

31
32 # Configuración inicial
33 np.random.seed(42)
34 plt.style.use('ggplot')
35
36 NumDatos=50; #Escribir el número de columnas a analizar de la DB
37
38
39 # Leer y preparar los datos (igual que en tu código original)
40 DataF1 = pd.read_csv("DataBase/BaseDatos.csv")
41
42 # Reducción de columnas Seleccionar 1 de cada 4 columnas usando iloc
43 ColDat=int(np.round(DataF1.shape[1]/NumDatos))
44 print(ColDat)
45 df_reducido = DataF1.iloc[:, ::ColDat] # Todos los renglones, 1
   columna cada 4
46
47
48 dummy_df = pd.get_dummies(DataF1, columns=["MOV_OUT"], dtype=int)
49 dummy_df.to_csv("DataBase/BaseDatos2.csv", index=False)
50 data = dummy_df
51
52 # Separar entradas (X) y salidas (y)
53 X = data.iloc[:, 1:800]
54 y = data.iloc[:, 800:]
55
56 # Verificar dimensiones
57 assert X.shape[1] == 799, f"Expected 800 input features, got {X.shape[1]}"
58 assert y.shape[1] == 11, f"Expected 11 output targets, got {y.shape[1]}"
59
60 # Dividir los datos en entrenamiento y prueba
61 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
   =0.2, random_state=42)
62

```

```

63 # Escalar los datos
64 scaler = StandardScaler()
65 X_train_scaled = scaler.fit_transform(X_train)
66 X_test_scaled = scaler.transform(X_test)
67
68
69 print("\n=== Análisis de Importancia de Variables con XGBoost ===")
70
71 # 1. XGBoost para importancia de características
72 print("\n1. Calculando importancia con XGBoost...")
73 start_time = time.time()
74
75 # Necesitamos convertir y a valores numéricos si no lo están ya
76 # Para problemas multioutput, necesitamos una estrategia diferente
77
78 # Opción 1: Para clasificación multioutput (si y es categórico)
79 from xgboost import XGBClassifier
80 from sklearn.multioutput import MultiOutputClassifier
81
82 # Configurar XGBoost
83 xgb = XGBClassifier(n_estimators=100, random_state=42, eval_metric='
      mlogloss', use_label_encoder=False)
84
85 # Usar MultiOutputClassifier similar a Random Forest
86 xgb_multi = MultiOutputClassifier(xgb)
87 xgb_multi.fit(X_train_scaled, y_train)
88
89 # Obtener importancia promedio de todas las salidas
90 importances_xgb = np.mean([estimator.feature_importances_ for
      estimator in xgb_multi.estimators_], axis=0)
91
92 # Crear DataFrame con las importancias
93 xgb_importance = pd.DataFrame({
94     'Feature': X.columns,
95     'Importance': importances_xgb
96 }).sort_values('Importance', ascending=False)

```

```

97
98 print(f"Tiempo de ejecución: {time.time() - start_time:.2f} segundos")
99 print(xgb_importance.head(50))  # Mostrar las 50 características más
    importantes
100
101 TOP_N = 50
102
103 plt.figure(figsize=(10, 50))  # Mucha altura, poco ancho
104 plt.barh(xgb_importance['Feature'].head(TOP_N)[::-1],
105          xgb_importance['Importance'].head(TOP_N)[::-1],
106          color='lightgreen')
107 plt.xlabel('Importancia')
108 plt.ylabel('Características')
109 plt.title(f'Top {TOP_N} características más importantes (XGBoost)')
110 plt.tight_layout()
111 plt.show()
112
113 import re
114
115 TOP_N = 50
116
117 # Obtener nombres de las top 20 características por importancia
118 top_features = xgb_importance['Feature'].head(TOP_N).tolist()
119
120 # Función para extraer el número al final de cada nombre
121 def obtener_numero(feature_name):
122     match = re.search(r'(\d+)$', feature_name)
123     return int(match.group(1)) if match else -1
124
125 # Ordenar por el número al final (cronológicamente)
126 top_features_ordenadas = sorted(top_features, key=obtener_numero)
127
128 # Imprimir en orden
129 print("\nTop 50 características más importantes (ordenadas cronoló
    gicamente):")
130 for i, f in enumerate(top_features_ordenadas, 1):

```



```

131     print(f"{i:>3}. {f}")
132
133 import pandas as pd
134 import re
135 import joblib
136
137 # --- Cargar base de datos original ---
138 ruta_original = "DataBase/BaseDatos.csv"
139 df = pd.read_csv(ruta_original)
140
141 # --- Guardar columna de clase MOV_OUT (se moverá al final) ---
142 clase = df["MOV_OUT"]
143
144 # --- Obtener las top 160 características por importancia ---
145 TOP_N = 50
146 top_features = xgb_importance['Feature'].head(TOP_N).tolist()
147
148 # --- Función para ordenar por número al final ---
149 def obtener_numero(feature_name):
150     match = re.search(r'(\d+)$', feature_name)
151     return int(match.group(1)) if match else -1
152
153 # --- Ordenar las top features cronológicamente ---
154 top_features_ordenadas = sorted(top_features, key=obtener_numero)
155
156 # --- Crear nuevo DataFrame con solo esas columnas y clase al final
157     ---
158 df_nuevo = df[top_features_ordenadas].copy()
159 df_nuevo["MOV_OUT"] = clase
160
161 # --- Guardar la nueva base de datos ---
162 ruta_salida = "DataBase/BaseDatos_Top50.csv"
163 df_nuevo.to_csv(ruta_salida, index=False)
164
165 # --- Guardar las top 50 características
166 joblib.dump(top_features_ordenadas, 'DataBase/top50_features.pkl')

```

```

166
167 print(f"\nNueva base de datos creada con las 50 características más
      importantes ordenadas cronológicamente.")
168 print(f"Guardada en: {ruta_salida}")
169
170 import pandas as pd
171 import numpy as np
172 from sklearn.model_selection import train_test_split
173 from sklearn.preprocessing import StandardScaler, LabelEncoder
174 from tensorflow.keras.models import Sequential
175 from tensorflow.keras.layers import Dense, Dropout
176 from tensorflow.keras.optimizers import Adam
177 from tensorflow.keras.utils import to_categorical
178 from tensorflow.keras.regularizers import l2
179 import joblib
180
181 # Cargar los datos
182 df = pd.read_csv('DataBase/BaseDatos_Top50.csv')
183 df.head()
184
185 df.values[:,1:-1]
186
187 # # Separar características y etiqueta
188 df2 = df.drop(["MOV_OUT"], axis=1)
189 X=df2.values[:,1:-1]
190 y = df['MOV_OUT'].values # Última columna
191
192 print(X)
193
194 # Preprocesamiento
195 # Codificar etiquetas
196 le = LabelEncoder()
197 y_encoded = le.fit_transform(y)
198 y_categorical = to_categorical(y_encoded)
199 joblib.dump(le, 'labelencoder.pkl')
200

```

```

201 # Verificar que tenemos 16 categorías
202 num_classes = len(le.classes_)
203 if num_classes != 11:
204     print(f"Advertencia: Se esperaban 11 categorías pero se
205         encontraron {num_classes}")
206
207 # Dividir datos con estratificación
208 X_train, X_test, y_train, y_test = train_test_split(
209     X,
210     y_categorical,          # <-- Se entrena con etiquetas one-hot
211     test_size=0.2,
212     random_state=42,
213     stratify=y_encoded      # <-- Estratificación usando etiquetas
214                             enteras
215 )
216
217 print(X_train.shape)
218 print(y_train.shape)
219
220 for i, clase in enumerate(le.classes_):
221     print(f"{i}: {clase}")
222
223 # Normalizar características
224 scaler = StandardScaler()
225 X_train = scaler.fit_transform(X_train)
226 X_test = scaler.transform(X_test)
227
228 # Crear el modelo de red neuronal
229 from tensorflow.keras.layers import LeakyReLU
230
231 model = Sequential([
232     Dense(256, kernel_regularizer=l2(0.001), input_shape=(48,)),
233     LeakyReLU(alpha=0.01),
234     Dropout(0.1),
235     Dense(128),

```

```

235     LeakyReLU(alpha=0.01),
236     Dropout(0.1),
237     Dense(64, kernel_regularizer=l2(0.001)),
238     LeakyReLU(alpha=0.01),
239     Dense(num_classes, activation='softmax')
240 ])
241
242 # Compilar el modelo
243 model.compile(optimizer=Adam(learning_rate=0.001),
244               loss='categorical_crossentropy',
245               metrics=['accuracy'])
246
247 from tensorflow.keras.callbacks import EarlyStopping
248
249 early_stop = EarlyStopping(monitor='val_loss', patience=10,
250                             restore_best_weights=True)
251
252 history = model.fit(X_train, y_train,
253                     epochs=100,
254                     batch_size=32,
255                     validation_split=0.1,
256                     callbacks=[early_stop],
257                     verbose=1)
258
259 # Evaluar el modelo
260 test_loss, test_acc = model.evaluate(X_test, y_test, verbose=0)
261 print(f'\nPrecisión en conjunto de prueba: {test_acc:.4f}')
262
263 # Guardar el modelo entrenado (opcional)
264 model.save('modelo_raw.h5')
265
266 # 1. GRÁFICO DE LA FUNCIÓN DE PÉRDIDA DURANTE EL ENTRENAMIENTO
267 import matplotlib.pyplot as plt
268 import seaborn as sns
269 plt.figure(figsize=(12, 6))

```

```

270 plt.plot(history.history['loss'], label='Training Loss')
271 plt.plot(history.history['val_loss'], label='Validation Loss')
272 plt.title('Training and Validation Loss')
273 plt.xlabel('Epoch')
274 plt.ylabel('Loss')
275 plt.legend()
276 plt.grid(True)
277 plt.show()
278
279 # PREDECIR CLASES PARA EL CONJUNTO DE PRUEBA
280
281 y_pred = model.predict(X_test)
282 y_pred_classes = np.argmax(y_pred, axis=1)
283 y_true_classes = np.argmax(y_test, axis=1)
284
285 # MATRIZ DE CONFUSIÓN
286 from sklearn.metrics import confusion_matrix, classification_report,
    f1_score
287 from tensorflow.keras.utils import to_categorical
288
289 conf_matrix = confusion_matrix(y_true_classes, y_pred_classes)
290 plt.figure(figsize=(12, 10))
291 sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
292 xticklabels=le.classes_, yticklabels=le.classes_)
293 plt.title('Matriz de Confusión')
294 plt.ylabel('Etiqueta Verdadera')
295 plt.xlabel('Etiqueta Predicha')
296 plt.xticks(rotation=45)
297 plt.yticks(rotation=0)
298 plt.show()
299
300 # MÉTRICAS DE CLASIFICACIÓN Y F1 SCORE
301
302 print("\nReporte de Clasificación:")
303 print(classification_report(y_true_classes, y_pred_classes,
    target_names=le.classes_))

```

```

304
305 # CALCULAR EL F1 SCORE MACRO Y WEIGHTED
306
307 f1_macro = f1_score(y_true_classes, y_pred_classes, average='macro')
308 f1_weighted = f1_score(y_true_classes, y_pred_classes, average='
    weighted')
309 print(f"\nF1-score Macro: {f1_macro:.4f}")
310 print(f"F1-score Weighted: {f1_weighted:.4f}")
311
312
313 # GUARDAR EL MODELO ENTRENADO Y SCALER
314
315 model.save('modelo_raw.h5')
316 joblib.dump(scaler, 'scaler.pkl')
317
318 # INFORMACIÓN ADICIONAL SOBRE EL MODELO
319 print("\nResumen del modelo:")
320 model.summary()

```

Listing 4.3: Algoritmo para Red Neuronal Feedforward por selección de características mediante XGBoost