Diseño de un sistema embebido para el reconocimiento automático de actividades físicas usando aprendizaje supervisado

Geenkel Miguel Coss Lara

2018

Resumen

El presente trabajo de tesis, describe el proceso de diseño y manufactura de un sistema personal de reconocimiento de actividades físicas que funciona en tiempo real. El sistema consiste de un dispositivo electrónico de recursos limitados que es portátil o *wearable* y un algoritmo inteligente. Para el reconocimiento de actividades se desarrolló un algoritmo clasificador basado en una Red Neuronal Artificial (RNA), para ello se realizó un proceso de recolección de muestras de entrenamiento, se extrajeron vectores de características de los datos recabados que luego se utilizaron para entrenar al algoritmo clasificador. Este algoritmo, fue posteriormente embebido en el dispositivo electrónico y se realizó un análisis de la eficiencia del mismo. Con base en los resultados obtenidos, se realizó un segundo diseño del dispositivo, con menor tamaño y prestaciones que el primero. Además, se mostró como puede adaptarse el proceso de desarrollo del clasificador de actividades para obtener un clasificador de caídas.

Índice general

Índice de figuras

Agradecimientos

Capítulo 1

Introducción

El diseño de metodologías para el monitoreo de la activación física es un tema que ha ido adquiriendo cada vez más importancia en los últimos años ya que es un factor importante en el combate a la obesidad. Una de las principales causas del sobrepeso y obesidad en una persona, es la falta de actividad física y por esta razón es relevante poder estimar cual es el nivel de activación de una persona durante periodos de tiempo definidos.

La obesidad, se ha estado incrementando a nivel mundial y constituye un problema de salud pública ya que suele generar otros problemas de salud que ponen en riesgo o disminuyen la calidad de vida humana. Nuestro país se encuentra en el primer lugar de porcentaje de casos de obesidad y sobrepeso en personas mayores a 15 años. El 72.5 % de la población nacional padece de sobrepeso u obesidad. El 33.3 % de esta población padece sobrepeso y el 39.2 % obesidad. Estos dos problemas son la principal causa de varias enfermedades, entre ellas, diabetes, cáncer y enfermedades cardiovasculares. Debido a esto, la demanda de tecnologías para el cuidado de la salud ha crecido en los últimos años. Gran cantidad de gente en la actualidad ya utiliza dispositivos o aplicaciones móviles que monitorean su actividad física diaria, calidad de sueño, ritmo cardíaco, entre otros.

La constante miniaturización de los circuitos integrados ha permitido añadir más poder computacional a dispositivos cada vez más pequeños, lo que ha ocasionado el rápido crecimiento de la industria de los dispositivos *wearables* o vestibles, conocidos de esta manera porque al ser pequeños pueden llevarse encima sin afectar las actividades cotidianas del usuario. Una de las principales aplicaciones que se le han dado a este tipo de dispositivos es el monitoreo de la activad física diaria.

En los dispositivos *wearables* que podemos encontrar en el mercado, el monitoreo de la actividad física generalmente consiste en contar el número de pasos que se han dado o medir el tiempo que el usuario estuvo caminando o corriendo.

Algunos de estos dispositivos también son capaces de detectar si se ha realizado alguna actividad además de caminar o correr, como montar en bicicleta o nadar.

Lo que se pretende con el monitoreo de la actividad física diaria utilizando este tipo de dispositivos, es llevar un control de la cantidad de energía o calorías que utilizamos diariamente con el fin de evitar el sedentarismo y sus consecuencias como el incremento de peso y la obesidad.

Para este propósito, conocer el tipo de actividad física que se realiza y durante cuánto tiempo puede ayudar a tener una mejor estimación del gasto energético. Una mejor medición de la actividad física podría permitir a un médico recomendar rutinas de ejercicio personalizadas o dietas que sean más eficientes con el consecuente beneficio de una mejor salud y por la tanto mejor calidad de vida. De esta manera, un dispositivo que sea capaz de reconocer de manera automática el tipo de actividad física, durante cuánto tiempo y la intensidad con la que se realiza obtendrá mejores resultados estimando el gasto energético. Otro beneficio que se obtendría de este tipo de dispositivo sería la verificación del cumplimiento de objetivos o metas de rutinas de ejercicios. El dispositivo podría llevar un registro del tiempo que se realiza cada actividad para verificar si se ha completado una meta o rutina personalizada.

En los últimos años el problema del reconocimiento automático de actividades ha sido abordado en una gran cantidad de trabajos que se pueden encontrar en la literatura. En ellos, se han propuesto técnicas para la clasificación de actividades, sin embargo, en la mayoría de estos trabajos, las técnicas son probadas de manera *offline* y no se analiza su implementación en tiempo real en un dispositivo vestible, el cual, debido a su tamaño, posee bajos recursos computacionales comparado con una computadora y además son alimentados con batería de muy baja capacidad, por lo que no resultaría práctico la implementación de un algoritmo complejo en un dispositivo de este tipo.

1.1. Objetivos

El objetivo general de este proyecto de tesis fue desarrollar e implementar un clasificador de actividades físicas utilizando técnicas de inteligencia artificial, en particular de aprendizaje supervisado, que funcione en tiempo real y además en un dispositivo *wearable* de bajos recursos computacionales.

Entre los objetivos particulares del proyecto están:

- Diseñar y fabricar un dispositivo *wearable*. Esto quiere decir que el dispositivo deberá ser pequeño, cómodo para el usuario, deberá funcionar con

batería y tendrá un consumo muy bajo de energía.

- Crear una base de datos de mediciones inerciales con diferentes personas realizando actividades físicas.
- Diseñar un sistema de clasificación de actividades físicas preciso y eficiente.
- Implementar el sistema de clasificación de actividades en el dispositivo *wearable* y realizar un análisis de su eficiencia.

1.2. Organización del documento

La organización del documento es como sigue: en el capítulo 2 se cubre el marco teórico y se introducen los conceptos clave para el desarrollo del proyecto; en el capítulo 3 se describe en detalle el estado de arte con una revisión de los trabajos en la literatura relacionados a la clasificación de actividades utilizando técnicas de reconocimiento de patrones; en el capítulo 4 se describe el diseño del dispositivo electrónico; en el capítulo 5 se presenta la metodología llevada a cabo y se describe el diseño del algoritmo clasificador de actividades, su implementación en tiempo real, tablas y gráficas con los resultados obtenidos en las pruebas *offline* del algoritmo y su funcionamiento en tiempo real; en el capítulo 6 se describe cómo se adaptó el clasificador para realizar detección de caídas; en el capítulo 7 se presenta una segundo versión del dispositivo electrónico. Finalmente, en el capítulo 8 se hace reflexión de lo obtenido y se presentan las conclusiones.

Capítulo 2

Marco teórico

El problema de diseñar un algoritmo para reconocer o identificar el tipo de actividad física que una persona se encuentra realizando, puede ser planteado como un problema de clasificación en el que un algoritmo determina cuál actividad física se está realizando de entre un conjunto de actividades. Por lo tanto, el algoritmo tendrá como entrada vectores de datos que describan la actividad y sean de utilidad para efectuar la discriminación entre las diferentes clases. La clase elegida, será la salida del algoritmo.

Un clasificador puede definirse como un algoritmo que asigna o etiqueta a una clase un conjunto de datos de entrada que son característicos de esa clase. De esta manera, un algoritmo clasificador, como ya se mencionó arriba, cuenta con una entrada y una salida. En el caso de la clasificación de actividades físicas, el clasificador tendrá como entrada información que describa la activad física que realiza y con la cual pueda discriminarse de otras actividades. Ejemplos de este tipo de información pueden ser: datos de la orientación y velocidad del movimiento de la persona que realiza la actividad, el ritmo cardíaco de la persona, respiración y algunos otros indicadores biométricos. La salida del algoritmo es la clase a la que pertenece la actividad física de acuerdo a la información de entrada.

En este trabajo, primero se procedió a identificar el conjunto de clases de salida, donde cada elemento de este conjunto, representa a una actividad física diferente. Una vez definido el conjunto de salida, se procedió a seleccionar los datos de entrada y la forma de obtenerlos.

En el caso particular del clasificador de actividades físicas, los datos de entrada del algoritmo se obtienen a través de sensores eléctricos. Estos se seleccionan de tal forma que proporcionen información relevante de la actividad cuando ésta se realiza y permiten que el algoritmo pueda reconocer y discriminar diferentes actividades. Cabe mencionar que un factor relevante en el óptimo rendimiento de un clasificador es la calidad de los datos de entrada. Esto es, una incorrecta

selección de los datos de entrada, influye significativamente en el rendimiento del clasificador al momento de asignar la clase. En este trabajo, para la obtención de los datos de entrada, se eligió un sensor inercial que se coloca en la muñeca. En otros trabajos reportados en la literatura, como se mostrará en la revisión del estado del arte, se han utilizado diversos sensores. Por ejemplo, cámaras digitales, micrófonos, sensores de posicionamiento global (GPS, por sus siglas en inglés), y algunos otros del mismo tipo.

Un sensor inercial, generalmente está conformado por tres tipos diferentes de sensores: acelerómetro, giroscopio y magnetómetro. En conjunto, estos sensores, proporcionan información sobre la aceleración, velocidad angular y fuerza magnética respectivamente, en cada uno de los 3 ejes x, y y z de un marco de referencia cartesiano. Si se coloca uno de estos sensores en alguna parte del cuerpo de una persona, se obtiene información relevante que puede ser usada para estimar el tipo de actividad que la persona está realizando.

En la literatura se encuentran clasificadores que utilizan algoritmos de inteligencia artificial, entre ellos, algoritmos de aprendizaje supervisado y no supervisado. En el desarrollo de los clasificadores por aprendizaje supervisado, existe una primera fase conocida como etapa de entrenamiento que es donde se ajustan los parámetros del modelo. En esta etapa, el clasificador utiliza un conjunto de datos etiquetados, conocido como datos de entrenamiento, para calcular los parámetros correctos del clasificador. Estos parámetros se calculan de tal manera que el clasificador acierte el mayor número de veces a la clase correcta del conjunto de datos de entrenamiento. A diferencia de los métodos de aprendizaje supervisado, los métodos de aprendizaje no supervisado, no necesitan que los datos de entrenamiento estén previamente etiquetados, sino que buscan encontrar patrones en esos datos para identificar las diferentes clases que puedan haber en ellos.

Para la clasificación de actividades físicas, los métodos de aprendizaje supervisado son los que podemos encontrar con mayor frecuencia en el estado del arte, ya que en este tipo de aplicaciones, la obtención de muestras clasificadas es relativamente fácil, y además, estos han demostrado tener un mayor rendimiento en la clasificación como veremos más adelante. A continuación se revisarán los fundamentos del funcionamiento de los algoritmos de aprendizaje supervisado y se describirán dos técnicas de este tipo que fueron utilizadas en este proyecto.

2.1. Aprendiza supervisado

Algoritmos clasificadores basados en técnicas de aprendizaje supervisado son los que más se han utilizado para resolver el problema de clasificación de actividades físicas. El aprendizaje supervisado es una rama de la inteligencia artificial y su objetivo es construir un modelo o función utilizando un conjunto de datos etiquetados conocido como conjunto de entrenamiento.

Existen dos tipos de algoritmos de aprendizaje supervisado: los de regresión y los de clasificación. Un algoritmo de regresión predice el valor continuo que le corresponde a una entrada, mientras que el algoritmo de clasificación asigna los datos de entrada a la clase a la que corresponden, por lo tanto su salida es discreta.

La entrada de un algoritmo clasificador son datos que describen o caracterizan lo que se desea clasificar y su salida es la clase a la que pertenecen esos datos. En el problema de clasificación de actividades físicas, los datos de entrada deben contener información acerca del movimiento de la persona y la salida deberá ser la clase ó la actividad que se estaba realizando al momento de obtener los datos del movimiento.

A la lista de datos de entrada que el algoritmo debe clasificar se le conoce como vector de características. El buen rendimiento del algoritmo depende en gran medida de qué tan bien describen las características del vector lo que se desea clasificar. Comúnmente, la selección de características se realiza de manera experimental, realizando pruebas con diferentes combinaciones para encontrar la adecuada con la que se obtienen los mejores resultados.

Para que un algoritmo pueda realizar la clasificación correctamente primero se lleva a cabo una etapa de entrenamiento o aprendizaje. El algoritmo de aprendizaje supervisado analiza ejemplos, es decir, vectores de características etiquetados con la clase a la que realmente pertenecen, y ajustan ciertos parámetros en el clasificador utilizando la información obtenida de estos ejemplos. Al conjunto de vectores de características etiquetados se le conoce como conjunto de entrenamiento. Posterior a esta fase, está la de clasificación en la que el algoritmo ya debe ser capaz de asignar una clase a vectores de característica de entrada no etiquetados utilizando los parámetros que almacenan la información del entrenamiento. La precisión o rendimiento del algoritmo es medido utilizando un conjunto de ejemplos de prueba que no fueron utilizados para el entrenamiento.

Algunos algoritmos, como el *K-Nearest Neighbor* no cuentan con una fase de entrenamiento previa a la clasificación como tal, sino que al momento de clasificar

un vector de características de entrada, analizan los datos de entrenamiento para decidir a que clase corresponde dicho vector.

El proceso para el desarrollo de un clasificador se puede resumir en los siguientes pasos:

1. Recolección de datos.
2. Preprocesamiento de datos.
3. Selección de características y construcción del conjunto de entrenamiento.
4. Entrenamiento del algoritmo.
5. Medición de rendimiento.

En la primera etapa se recolectan datos que describen lo que se desean clasificar y se construye una base de datos con muestras etiquetadas que servirán para entrenar el algoritmo y probar su rendimiento. En la siguiente etapa se realiza un preprocesamiento de los datos recolectados, que puede ser una selección de muestras relevantes, filtrado de los datos o la normalización de los mismos, etc. Después se seleccionan los elementos que conforman el vector de características y se construye el conjunto de entrenamiento que contiene vectores etiquetados de todas las clases. En la siguiente etapa se utiliza un algoritmo para entrenar el clasificador, se realizan pruebas clasificando vectores etiquetados y se utilizan diferentes medidas para determinar el rendimiento del clasificador. Si el resultado obtenido no es satisfactorio se regresa a cualquier etapa previa del proceso para hacer ajustes hasta encontrar un clasificador óptimo.

A continuación se describirá el funcionamiento de dos métodos de aprendizaje supervisado que fueron utilizadas en este proyecto para construir un clasificador de actividades físicas: K Vecinos más Cercanos y Redes Neuronales Artificiales.

2.2. K Vecinos más Cercanos (*K-NN*)

El algoritmo K-Vecinos más Cercanos(*K-NN* por sus siglas en inglés) es una técnica de clasificación de aprendizaje supervisado. Su funcionamiento se describe a continuación: *K-NN* clasifica una nueva muestra $\boldsymbol{p} = [p_1, p_2, \ldots, p_N]$ en la clase más frecuente a la que pertenecen las K muestras más cercanas en el conjunto de entrenamiento. El algoritmo, por lo tanto, utiliza una función para calcular la distancia entre los vectores de características de entrenamiento y el de la muestra

a clasificar. La función de distancia más utilizada es la distancia euclidiana dada por la siguiente ecuación:

$$d(\boldsymbol{p}, \boldsymbol{q}) = \sqrt{\sum_{i=1}^{N}(p_i - q_i)^2} \tag{2.1}$$

donde $\boldsymbol{p}$ es la muestra que se desea clasificar, $\boldsymbol{q}$ es una muestra en el conjunto de entrenamiento, y p_i y q_i son las características de $\boldsymbol{p}$ y $\boldsymbol{q} = [q_1, q_2, \ldots, q_N]$ respectivamente.

El ejemplo más sencillo, cuando K=1, la nueva muestra P se clasifica en la clase a la que pertenece la muestra etiquetada $\boldsymbol{q}$ en el conjunto de entrenamiento que tenga la menor distancia euclidiana a $\boldsymbol{p}$.

Este método solamente cuenta con un parámetro, que es el número K y su valor no se elige según los datos de entrenamiento si no que se selecciona de manera experimental.

Como se puede notar, este algoritmo no cuenta con una fase de entrenamiento previa en la que se encapsula la información de entrenamiento, si no que debe siempre tener disponible el conjunto de entrenamiento al realizar una clasificación.

Durante una clasificación, el algoritmo busca en todo el conjunto de entrenamiento de n muestras los k-vecinos más cercanos. En la implementación simple del algoritmo por fuerza bruta, la complejidad en tiempo sería de $O(kn)$ y en memoria $O(n)$. Esta es la principal desventaja de utilizar este algoritmo para clasificar, sobre todo si tenemos grandes cantidades de datos de entrenamiento. Su ventaja es su simpleza y facilidad de implementar.

2.3. Redes Neuronales Artificiales

La red neuronal artificial (RNA) es un algoritmo computacional que simula el sistema de conexiones neuronales del sistema nervioso de los seres vivos, en donde al percibir un estímulo, cada neurona y por lo tanto, la red neuronal, producen una respuesta.

Este sistema posee la capacidad de aprender y ajustarse para modelar diferentes problemas, y es lo que los hace muy útiles en diferentes áreas. En este proyecto las RNAs fueron utilizadas para realizar la clasificación de actividades físicas. En este caso, el estímulo de entrada es la información del movimiento de una persona y la respuesta es la actividad física que se realiza.

Una RNA está formado por un conjunto de unidades de procesamiento llamadas neuronas o perceptrón. Cada perceptrón en una red, está compuesto de

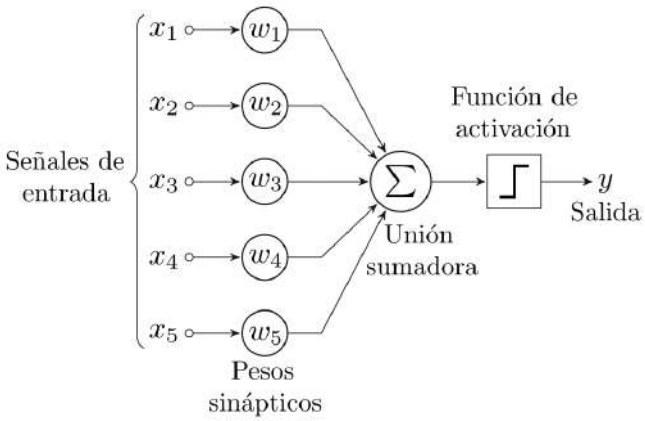

Figura 2.1: Diagrama del Perceptrón.

varias entradas, a las cuales se les asigna un peso, una función de activación y una salida, la cual puede funcionar como entrada de otra neurona o como una salida de la red. En la figura 2.1 se aprecia el diagrama de un perceptrón.

La salida de una neurona está dada por la evaluación en la función de activación del resultado de la suma de las entradas multiplicadas por sus respectivos pesos. La siguiente ecuación representa la salida del perceptrón:

$$y(\boldsymbol{x}, \boldsymbol{w}) = f\left(\sum_{j=1}^{M} w_j x_j\right) \tag{2.2}$$

en donde $\boldsymbol{x} = [x_1, x_2, \ldots, x_M]$ es el vector de características de entrada $\boldsymbol{w} = [w_1, w_2, \ldots, w_M]$ es el vector de pesos y $f(\cdot)$ es la función de activación. Esta es una función diferenciable no lineal con asíntotas horizontales que generalmente es la función Sigmoide (2.3) o la función tangente hiperbólica (2.4). En las figuras 2.2 y 2.3 se puede apreciar el comportamiento de ambas funciones.

$$f(y) = \frac{1}{1 + \exp(-y)} \tag{2.3}$$

$$f(y) = tanh(y) = \frac{\exp(y) - \exp(-a)}{\exp(a) + \exp(-a)} \tag{2.4}$$

2.3.1. Redes Multicapa de Propagación Hacia Adelante

Las redes multicapa de propagación hacia adelante son un tipo de RNA con varias capas de neuronas en donde cada una de las salidas de las neuronas de una capa se conectan a todas las neuronas de la siguiente capa. Este tipo de redes también son conocidas como perceptrón multicapa, son las más utilizadas ya que en la práctica se ha encontrado que poseen un buen rendimiento en problemas de

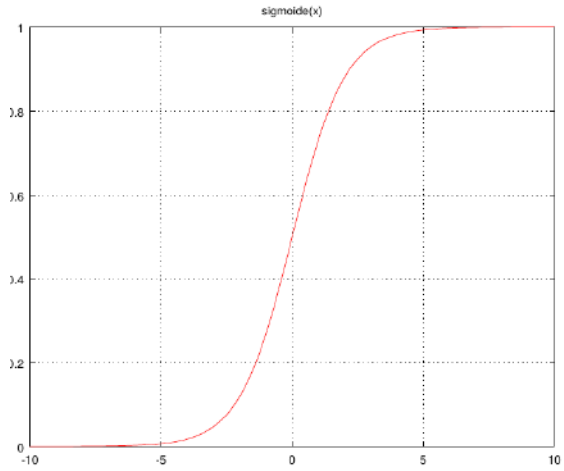

Figura 2.2: Gráfica de la función Sigmoide

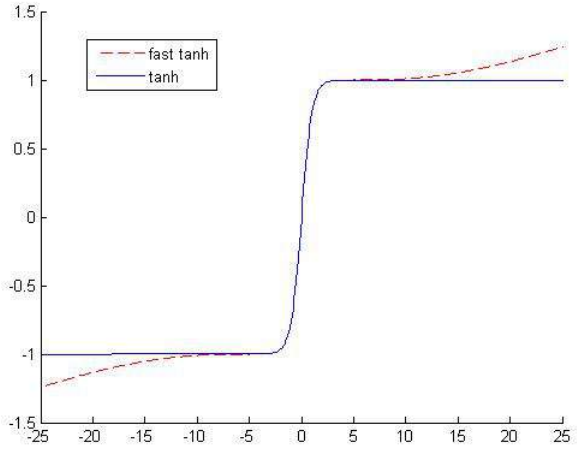

Figura 2.3: Gráfica de la función Tangente Hiperbólica

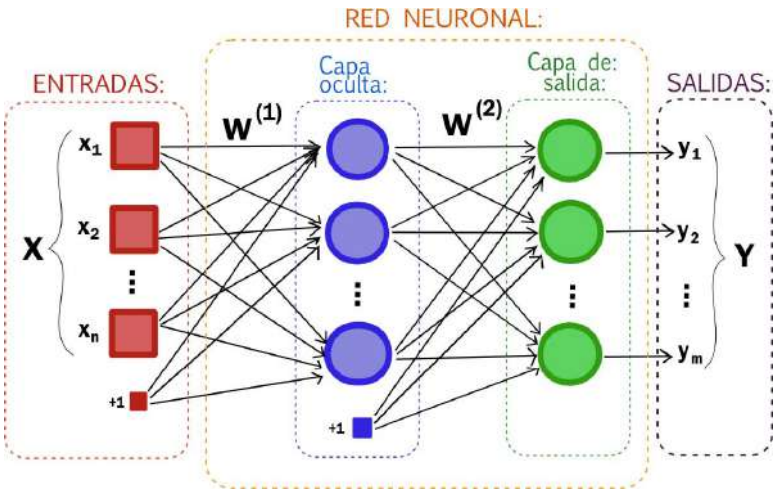

Figura 2.4: Red Neuronal con topología de propagación hacia adelante.

clasificación y reconocimiento de patrones. A diferencia del perceptrón simple que solamente sirve para resolver problemas linealmente separables, se ha demostrado que con al menos dos capas de neuronas se puede aproximar cualquier función. En la figura 2.4 se puede ver un ejemplo de este tipo de topología de red.

2.3.2. Algoritmo de aprendizaje de propagación hacia atrás o retropropagación

La retropropagación es el algoritmo de aprendizaje utilizado para entrenar una RNA de tipo perceptrón multicapa. El entrenamiento en una RNA se refiere al proceso que se lleva acabo para encontrar los pesos correctos para que la red modele un problema con el mínimo error, que en nuestro caso será para que la red reconozca o clasifique actividades físicas. Para ello, se lleva a cabo un algoritmo de optimización que va modificando los pesos de la red mientras se reduce el error de clasificación de los ejemplos en el conjunto de entrenamiento.

Este algoritmo se puede resumir de manera breve en los siguientes pasos:

1. Se inicializa el vector de pesos $\boldsymbol{w}$ de la red neuronal con valores aleatorios.

2. Se propaga un vector de características del conjunto de entrenamiento $\boldsymbol{x}_n$ a través de la red para encontrar la salida de todas las neuronas.

3. Para cada neurona (hay k neuronas) en la capa se salida se calcula su término de error δ_k:

$$\delta_k = y_k - t_k \tag{2.5}$$

   en donde y_k es la salida de la neurona en la capa de salida y t_k el valor objetivo o esperado de salida para el ejemplo $\boldsymbol{x}_n$.

4. Se propaga el error hacia atrás calculando δ_j para cada neurona en las capas ocultas:

$$a_j = \sum_{i=0}^{D} w_{ji} x_i \tag{2.6}$$

$$\delta_j = h'(a_j) \sum w_{kj} \delta_k \tag{2.7}$$

   en donde w_{ji} es el peso de la $j-$ésima neurona de una capa oculta asignado a su entrada x_i, que puede ser un elemento del vector de características de entrada o la salida de una neurona de la capa anterior. h' es la derivada de la función de activación, δ_k es el término de error de la $k-$ésima neurona de la siguiente capa y w_{kj} es el peso asignado a la salida de la $j-$ésima neurona.

5. Se actualizan los pesos de las neuronas de la capa de entrada con $\triangle w_{ji} = \eta \delta_j x_i$ para pesos de neuronas en capas ocultas o $\triangle w_i = \eta \delta_h x_i$ para neuronas en la capa de salida. η es el factor de aprendizaje y se selecciona de manera empírica. Generalmente tiene un valor de entre 0 y 1.

6. Se repite lo mismo para todos los ejemplos de entrenamiento $\boldsymbol{x}_n$. A esto se le conoce como época.

7. Se repiten varias épocas mientras el error de clasificación se reduzca.

Al finalizar el entrenamiento, la información de los ejemplos queda encapsulada en los pesos de las neuronas y la red puede utilizarse para clasificar nuevas muestras. La desventaja de este método es que el algoritmo no necesariamente encuentra el mínimo global de la función de error de la RNA, sin embargo, en la mayoría de los casos se obtendrán buenos resultados, aún sin encontrar el mínimo global. Solamente será necesario realizar varias corridas de este algoritmo, medir el rendimiento de cada una y quedarse con la que se obtengan los mejores resultados.

A diferencia del algoritmo *K-NN* la complejidad de un RNA no depende del conjunto de entrenamiento si no del número de neuronas que compongan la RNA. Su principal desventaja es la complejidad de tiempo del algoritmo de entrenamiento, sin embargo el tiempo de clasificación es relativamente más corto lo que lo hace ideal para aplicaciones en tiempo real. En cuanto a la complejidad en memoria, el algoritmo solamente necesita tener disponible los pesos de cada neurona en la red.

Capítulo 3

Estado del arte

En esta sección se realiza una revisión resumida de los trabajos relacionados a la clasificación automática de actividades que se encuentran en la literatura. Primero se realiza una revisión general y al final se identifican y describen 3 trabajos en donde se analiza la implementación de los algoritmos de clasificación de actividades en tiempo real en dispositivos *wearables*.

La miniaturización de los circuitos electrónicos que ha permitido el desarrollo de sensores cada vez más pequeños y la incorporación de mayor poder computacional en dispositivos portátiles, ha potenciado el interés en aplicaciones como la detección automática de actividades. Debido a esto, en la literatura podemos encontrar actualmente gran cantidad de trabajos en los que se aborda este tema.

Las principales aplicaciones de un sistema de reconocimiento automático de actividades se encuentran en el área del cuidado de la salud. Como se ha mencionado, con un sistema de detección de actividades se puede obtener retroalimentación de la actividad física diaria de una persona o de las rutinas de ejercicios que sigue con el fin de favorecer su activación física y mejorar su calidad de vida. En [8] se desarrolló un sistema de detección de actividades para el cuidado del adulto mayor utilizando cámaras de video, el cual puede detectar seis actividades anormales que afectan la salud, entre ellas caerse o vomitar. En [5] se describe una plataforma para el monitoreo de pacientes de diabetes, en donde se incluye un sistema de reconocimiento de actividades para el análisis del comportamiento de los pacientes. Otras aplicaciones se encuentran en el área de seguridad. En [16] se implementa un sistema de detección de actividades anormales, el cual puede aplicarse en la detección de comportamientos terroristas en zonas de alta seguridad o también para detectar comportamientos raros para el diagnóstico de enfermedades como el Alzheimer o Parkinson. Otra aplicación se encuentra en el área laboral, en donde un sistema de este tipo puede utilizarse para monitorear las actividades de los empleados y asegurar que realicen el trabajo asignado. En

Grupo	Actividades
Ambulación	Caminar, correr, permanecer sentado o de pie, subir o bajar escaleras.
Transporte	Usar el autobús, montar bicicleta, manejar.
Vida diara	Comer, beber, ver TV, lavarse los dientes, usar la PC, leer un libro.
Ejercicio	Abdominales, lagartijas, spinning.
Social	Platicar.

Cuadro 3.1: Tipos de actividades que se han detectado (tomado del artículo [Lara2013])

[15] se propone un método para el reconocimiento de actividades de enfermeras en un ambiente de laboratorio utilizando sensores acelerómetros y de posición.

Las actividades que el sistema de clasificación reconocerá dependen de la aplicación que tendrá el sistema final. En el artículo [11] se realiza una revisión de los artículos científicos relacionados al tema de la clasificación de actividades utilizando sensores *wearables*. La tabla 3.1 fue tomada de este documento y en ella se encuentra una lista resumida de los tipos de actividades que se han intentado clasificar en la literatura junto con algunos ejemplos.

A su vez, dependiendo de qué actividades se desean detectar, se seleccionan los sensores adecuados que extraerán información del movimiento de los usuarios y que servirá como datos de entrada al algoritmo clasificador. En la literatura podemos encontrar que los sensores más comunes en la detección de actividades son los sensores inerciales, en especial los acelerómetros, ya que se ha visto que son los que proporcionan datos más relevante debido a que esta información está directamente relacionada al movimiento del usuario. Además de estos sensores, también se han utilizado micrófonos, como se reporta en el trabajo [3], en donde se desarrolló un dispositivo con varios sensores, entre ellos, un micrófono que se utiliza para reconocer actividades como platicar o ver la televisión. En el trabajo [13] se utiliza un sensor de posición GPS para reconocer actividades como dormir, trabajar, viajar en auto, ir de visita. En el dispositivo desarrollado en [3] se incorporaron sensores de temperatura, humedad y luz ambiental para identificar el contexto en el que se realiza la actividad, por ejemplo, si esta se realiza día o de noche, en el interior de un edificio o en exteriores. Por último, en el trabajo de Bingbing et al. [2] se utilizaron datos de color RGB y de profundidad de una cámara para construir una base de datos llamada *RGBD-HuDaAct*, la cual se utilizó para entrenar un clasificador de actividades de la vida diaria realizadas en casa como comer, dormir, vestirse o hacer una llamada por teléfono.

Debido a que en este proyecto se pretende desarrollar un clasificador de actividades físicas relacionadas a rutinas de ejercicios, los sensores inerciales resultan ser los más adecuados. Estos sensores a diferencia de una cámara de video, proporcionan información directamente relacionada al movimiento del usuario en pocos valores por lo que no se necesita muchos recursos computacionales para procesarlos. Un sistema de detección de actividades con cámaras de video tendría un costo mucho mayor debido a los recursos necesarios para procesar el video. Otro desventaja es que el usuario solamente podría realizar las actividades dentro del campo de visión de la cámara, lo cual no es práctico para actividades como correr o caminar y además el usuario podría sentir que se invade su privacidad. Al existir modelos muy pequeños de sensores inerciales y de bajo consumo de energía, pueden incorporarse a dispositivos *wearables*. En cuanto a los sensores GPS, aún no son lo suficientemente pequeños y tienen un consumo alto de energía como para integrarlos a un dispositivo *wearable* sin afectar la autonomía del mismo. Por último, debido al tipo de actividades que se detectarán, los sensores ambientales no proporcionarían información útil para la clasificación. Sensores biométricos como los sensores de frecuencia cardíaca o musculares podría ser de utilidad para nuestra aplicación, sin embargo, el alcance del proyecto solo abarca el uso de sensores inerciales.

En el área de la detección de actividades, utilizar un método determinista que mapee los datos de los sensores con su correspondiente actividad resulta totalmente impráctico. El número de combinaciones de datos de entrada y actividades es sumamente grande. Debido a esto, se recurre a métodos de aprendizaje máquina. En la literatura podemos encontrar que la mayoría de las técnicas utilizadas son de aprendizaje supervisado y en algunos trabajos se ha usado aprendizaje semi-supervisado. Por ejemplo, en [9] se utilizan redes neuronales y regresión logística para clasificar actividades de la vida diaria utilizando el acelerómetro de un teléfono celular. En [6] se probaron Redes Bayesianas, K vecinos más cercanos y árboles de decisión. En [7] se utilizaron Modelos Ocultos de Markov para detectar actividades con datos provenientes de 5 sensores colocados en diferentes partes del cuerpo. En [4] se propone un método semi-supervisado llamado *En-Co-training* que consiste en entrenar un clasificador con datos etiquetados y reentrenarlo posteriormente con nuevos datos etiquetados con el clasificador entrenado previamente. Por último, aunque menos común, también se ha utilizado aprendizaje no supervisado como se demuestra en [10], en donde prueban este tipo de métodos para compensar las desventajas del método supervisado, entre ellas, la dificultad de añadir nuevas clases y la poca flexibilidad que puede llegar

a tener.

Otro aspecto a analizar de los trabajos en esta área es la arquitectura del sistema de detección de actividades, la cual está compuesta por los sensores y los dispositivos que realicen el procesado de los datos, la clasificación de estos y el despliegue de los resultados. También podemos identificar los protocolos de comunicación entre los distintos dispositivos que conforman el sistema ó si la clasificación se lleva a cabo en tiempo real o de manera *offline*. En la mayoría de los trabajos no se toma en cuenta la aplicación del sistema de clasificación en tiempo real y solamente se realizan pruebas y se evalúan los algoritmos *offline* en una computadora. Por lo tanto, no se analiza su complejidad, consumo de recursos ó su implementación en un sistema embebido o dispositivo *wearable*. En algunos trabajos, los dispositivos *wearables* solamente se utilizan para realizar la recolección de datos de sensores y estos son enviados utilizando algún protocolo de comunicación inalámbrica a un teléfono celular o computadora. Los datos pueden enviarse crudos, filtrados o una vez que ya se hayan extraído las características, para ser luego procesados o clasificados por otro dispositivo con mayor poder computacional, como una PC, computadora portátil o un teléfono celular que después puede desplegar los resultados.

Por ejemplo, en [12] se implementó un sistema de clasificación de actividades cuya arquitectura consta de un dispositivo *wearable* que recolecta datos de sensores, un teléfono móvil que recibe estos datos en crudo a través de comunicación inalámbrica, extrae las características y ejecuta los algoritmos de clasificación. Los resultados son enviados a través de Internet a un servidor y finalmente, estos pueden ser visualizados en tiempo real utilizando una página web. En la figura **??**, tomada de ese artículo, se ilustra esta arquitectura.

Debido a que nuestra aplicación es la detección de actividad física, un usuario que sale a hacer ejercicio corriendo o realiza rutinas en el gimnasio desea conocer el tiempo que ha realizado cada ejercicio y las calorías que ha utilizado para determinar si ha alcanzado el objetivo de una rutina personalizada, por lo tanto, la detección de actividades se espera que sea en tiempo real. Utilizar una computadora portátil o PC no resultaría práctico en este caso y llevar consigo un teléfono móvil podría ser estorboso para realizar ciertos ejercicios físicos. Debido a esto, lo ideal sería que un dispositivo *wearable* realice todas las tareas implicadas en la detección de las actividades: la recolección de datos, filtrado y extracción de características, clasificación de la actividad y desplegado de resultados. Esto representa un reto, ya que los dispositivos de este tipo, generalmente cuenta poco poder computacional debido a su tamaño y al funcionar con baterías de baja

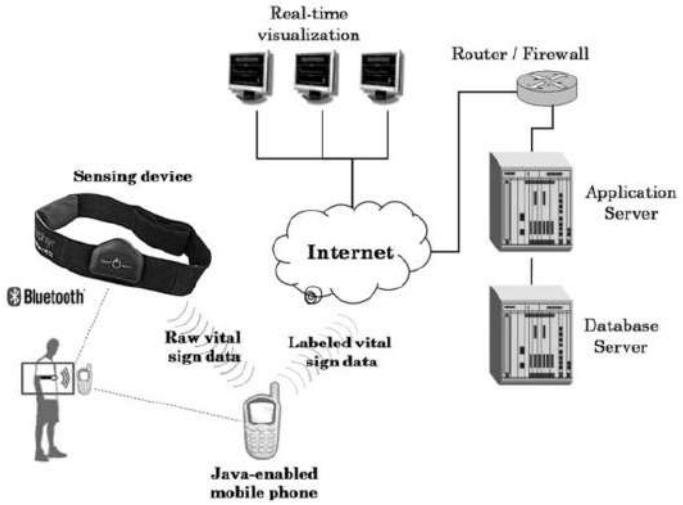

Figura 3.1: Arquitectura del sistema de clasificación de actividades **Centinela** [12]

capacidad se debe procurar utilizar la menor cantidad de recursos para alargar la autonomía del dispositivo. Por lo tanto, es de gran importancia analizar la complejidad en tiempo y memoria de los algoritmos de extracción de características y clasificación, y con base a este análisis encontrar la mejor combinación de precisión en clasificación y uso de recursos computacionales.

Muy pocos de los trabajos en la literatura han tomado en cuenta estos aspectos y han implementado un clasificador de actividades que funcione en tiempo real en un dispositivo *wearable*. A continuación se analizarán 3 artículos en donde los algoritmos de recolección de muestras, extracción de características y clasificación están embebidos en un dispositivo *wearable*.

3.1. Mobile Sensing Platform (MSP) [3]

En el trabajo de Choudury *et al.* [3] se describe un dispositivo móvil desarrollado por ellos llamado *Mobile Sensing Platform* (**MSP**) en su segunda versión, el cual incluye siete diferentes sensores: micrófono, luz ambiental, temperatura, luz infrarroja, acelerómetro de tres ejes, magnetómetro y de humedad. La electrónica tiene unas dimensiones de 3.6 cm por 5.1 cm y el dispositivo se coloca en la cintura del usuario. En la figura 3.2 se presentan algunas imágenes de este dispositivo. El **MSP** cuenta con un microcontrolador *Atmega128*, que realiza la lectura de los sensores, y con una tarjeta *iMote2* de *Intel* que funciona con un procesador de 416 MHz. También cuenta con comunicación *Bluetooth* y una ranura para memorias *micro SD*. El dispositivo es alimentado por una batería de 1800 mAh, la cual

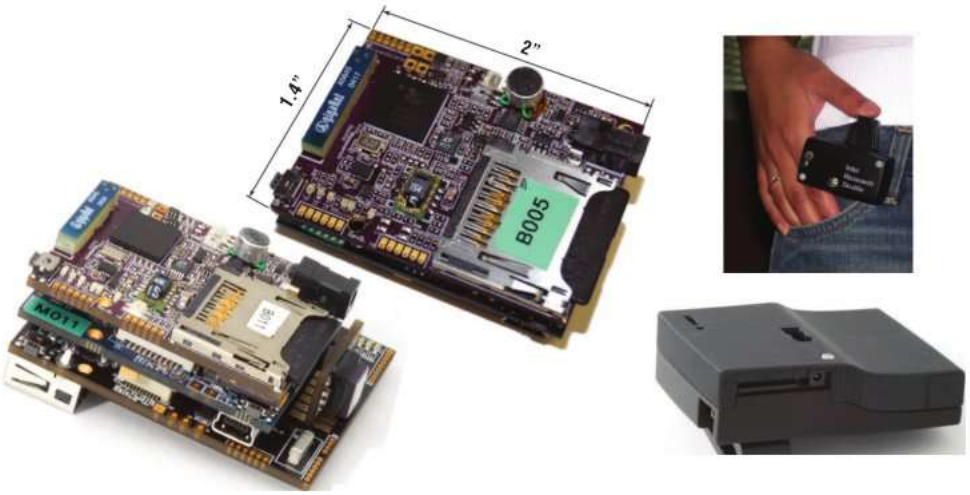

Figura 3.2: Mobile Sensing Platform [3]

le da una autonomía de 10 a 20 horas. Para clasificar actividades implementaron 3 algoritmos con los que realizaron pruebas, dos de ellos utilizando aprendizaje supervisado y el otro utiliza aprendizaje semi-supervisado. Las actividades con las que realizaron las pruebas son: caminar, estar sentado o parado, utilizar escaleras o elevador y lavarse los dientes. En el artículo se reporta haber logrado una precisión de 93.8 % en la clasificación.

3.2. eWatch [14]

En el artículo de Maurer *et. al* [14] se describe un dispositivo *wearable* llamado **eWatch**. Este cuenta con sensores de temperatura, de luz, micrófono y acelerómetro. El dispositivo se coloca en la muñeca del usuario y tiene unas dimensiones de 50mm x 48mm x 17.5mm. En la figura 3.3 se pueden ver fotos del dispositivo y la electrónica. Este es controlado por un CPU Philips de 60Mhz e integra un módulo Bluetooth para comunicación inalámbrica con un teléfono celular. Se alimenta con una batería recargable de Litio-polímero de 700 mAh, la cual permite al dispositivo funcionar por 56 horas. Los sensores de temperatura, luz y el micrófono son utilizados para un algoritmo de detección de lugares visitados. Para el clasificador de actividades se utilizan los datos del acelerómetro y del sensor de luz. El acelerómetro es de dos ejes, tiene un rango de $\pm 2G$, 8 bits de resolución y 20 Hz de frecuencia de muestreo. Con el fin de encontrar el mejor balance entre complejidad y precisión, realizaron diferentes pruebas colocando el **eWatch** en diferentes partes del cuerpo y se probaron con diferentes vectores de características. El trabajo reportado con respecto a la clasificación de actividades

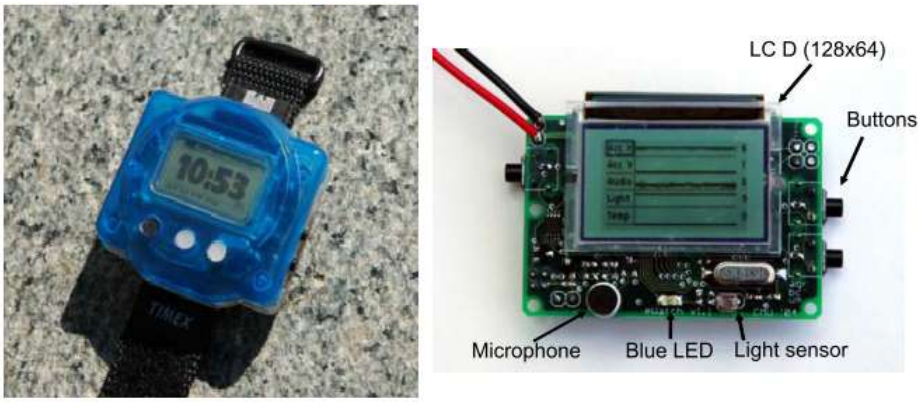

Figura 3.3: Dispositivo **eWatch** [14]

se resume brevemente en los siguientes puntos:

- Se recolectaron datos de los sensores de aceleración y luz en 6 voluntarios realizando las siguientes actividades: correr, caminar, estar de pie, estar sentado, subir y bajar escaleras. Los valores de los sensores se partieron en ventanas de tiempo sobre las que se calculan las características.

- Se calcularon características tomando individualmente los ejes X y Y del acelerómetro y combinando ambos utilizando la magnitud del vector. Todas las características utilizadas están en el dominio del tiempo. El calculo de las características fue implementado en el **eWatch** y se midieron los tiempos de calculo de cada una de ellas. Entre las características que se probaron están: media, desviación estándar, varianza e histograma acumulado. Todas se calcularon sobre ventanas de tiempo de 4 segundos (80 ejemplos).

- Se realizaron pruebas utilizando diferentes combinaciones de características encontradas con un método de selección de principales características.

- Se evaluaron varios métodos de clasificación: Árboles de decisión, K-NN, Naive-Bayes y Red Bayesiana. Se midió el tiempo de clasificación y la precisión de cada uno.

- Se llegó a la conclusión de que el método de Árbol de decisión tiene el mejor balance de complejidad y precisión. Además, se obtienen los mejores resultados al colocar el **eWatch** en la muñeca.

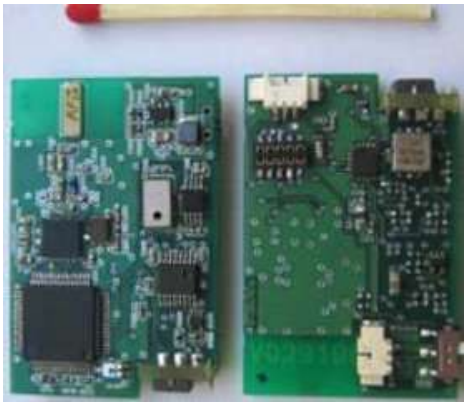

Figura 3.4: Multi-Sensor Context Recognition Platform [1]

3.3. Multi-Sensor Context Recognition Platform [1]

En el artículo de Bharatula *et. al* [1] se presenta un dispositivo *wearable* de bajo consumo de energía, el cual incorpora acelerómetros, un sensor de luz y un micrófono. La electrónica (figura 3.4) mide 4.15 cm x 2.75 cm y es controlador por un MSP430F1611 de *Texas Instruments* y además cuenta con comunicación inalámbrica a través de un transceptor NRF2401E. Una batería de 130 mAh alimenta el circuito y se reporta que tiene una duración de 129 horas.

Se implementó un clasificador embebido basado en árboles de decisión para reconocer 6 actividades: caminar, beber de una taza, dormir, escribir, utilizar una computadora y estar sentado o quieto. Para la obtención de características las señales de aceleración en los 3 ejes y de luz son muestreados a 32 Hz y del micrófono a 4096 Hz. Se toma una ventana de datos de 4 segundos con una superposición de 75 %, obteniendo de esta manera una clasificación por segundo. De las pruebas realizadas se obtuvo que el clasificador tiene una precisión de 87 %.

Capítulo 4

Diseño del dispositivo

En este trabajo de tesis se diseñó y fabricó un dispositivo que fue utilizado tanto para realizar la recolección de muestras para el armado de una base de datos de actividades, como para la implementación de un clasificador de actividades físicas en tiempo real.

El dispositivo fue pensado para usarse como un dispositivo "wearable", por lo tanto, este debe ser pequeño, ligero y debe poder llevarse encima sin afectar las actividades normales del usuario. Considerando esto, la selección de los componentes que se incluyeron en el diseño del dispositivo se realizó tomando en cuenta las siguientes características:

- **Tamaño del empaquetado del componente.** Generalmente, el mismo componente se puede encontrar en diferentes tamaños o encapsulados. Durante la selección se trato de escoger los tamaños más pequeños pero teniendo en cuenta las restricciones que se tenían debido al método de fabricación de placas de placas de circuito impreso y de montaje de componentes.

- **Voltaje de trabajo y consumo de energía.** Debido a las restricciones que debe cumplir el dispositivo se decidió que este funcionara con una batería recargable de Ion-Litio, las cuales tienen un voltaje de 3.7V, por tanto, se escogió un voltaje de trabajo estándar para los componentes de 3.3 Volts. Los componentes seleccionados soportan este voltaje de alimentación. Además, se trató de escoger aquellos que tuvieran un consumo muy bajo de energía con el fin de aumentar la autonomía del dispositivo.

- **Componentes externos necesarios.** El diseño del dispositivo se realizó con circuitos integrados que requieren de un mínimo de componentes externos (componentes pasivos por lo general) para trabajar, con el fin de mantener reducido el tamaño del dispositivo.

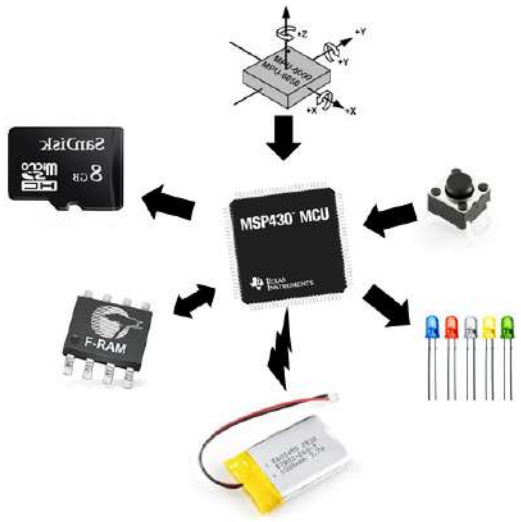

Figura 4.1: Componentes del sistema.

- **Precio.** Por último, se buscó un balance entre las características anteriores y el costo de los componentes.

El dispositivo fabricado en este proyecto, como se mencionó, está orientado a la recolección de muestras de entrenamiento, por tanto, se añadieron al diseño los componentes necesarios para este propósito y para la fácil transferencia de las muestras a una computadora para la etapa de entrenamiento del algoritmo de clasificación. Por este motivo, el tamaño y consumo de energía del dispositivo pudiera no ser lo óptimo para un dispositivo *wearable*. En el capítulo 9 se muestra una segunda versión del dispositivo, el cual si cumple con las restricciones de un dispositivo de este tipo.

Para entender mejor el diseño del dispositivo electrónico, este se dividió en los siguientes 6 módulos: Microcontrolador, Sensor inercial, Memoria F-RAM, Memoria Micro SD, Botones y diodos LED, y alimentación. En la figura 4.1 se ilustran los módulos que componen el sistema. A continuación se describirán por separado cada uno de estos.

Microcontrolador

Figura 4.2: Microcontrolador MSP430F5529 de *Texas Instruments*.

El microcontrolador que se añadió al diseño fue el MSP430F5529 de *Texas Instruments*, principalmente debido a que tiene un consumo muy bajo de energía, lo que permite una mayor autonomía del dispositivo, que es una característica de gran importancia en los dispositivos *wearables*. El MSP430F5529 es un microcontrolador con arquitectura de 16 bits y el reloj del procesador tiene una velocidad máxima de 25 MHz. Una de sus principales características es que permite configurar de manera flexible y en tiempo de ejecución la velocidad de las fuentes del reloj para el procesador y los periféricos y además éste puede operar en varios modos de energía. Con estas características se puede obtener un rendimiento y consumo de energía óptimo según la tarea que se realice. El microcontrolador puede configurarse para operar a baja velocidad para ahorrar energía o de manera análoga, puede configurarse para funcionar rápido en tareas más demandantes. Mientas no se esté realizando ninguna tarea, el MSP430 puede configurar al procesador en un estado inactivo para reducir el consumo de energía. Además, este cuenta con módulos de comunicación SPI e I2C necesarios para controlar el sensor, la memoria FRAM y la memoria Micro SD.

Entre otras características, el MSP430F5529 cuenta con 128 KB de memoria Flash, que es donde reside el programa, y 8 KB de memoria RAM para los datos. Otro punto a remarcar del MSP430 es que no cuenta con un módulo de hardware para operaciones de punto flotante, por lo que la ejecución de este tipo de operaciones exigirán una cantidad importante de recursos computacionales. Como podemos notar, los recursos con los que cuenta el MSP430F5529 son bastante limitados, lo que también limita la complejidad de los algoritmos que pueden ejecutarse en él. A pesar de estas desventajas, su muy bajo consumo de energía, tamaño y costo hacen que el microcontrolador MSP430 sea ideal para dispositivos *wearables*.

Cuando se programa un microcontrolador que es parte de un dispositivo *wearable*, los cuales funcionan con baterías, además de que se debe estar consciente

de las limitaciones de memoria y procesamiento, siempre se debe tratar de que los algoritmos se ejecuten en el menor tiempo posible. Esto con el fin de mantener el mayor tiempo al procesador en estado inactivo, reduciendo el consumo de la energía de la batería y aumentando de esta forma la autonomía del dispositivo. En el caso del MSP430F5529, estando en modo inactivo se puede logar un consumo desde 2.9 μA hasta un máximo de 92 μA. En modo activo tiene un consumo aproximado de 290 μA/MHz.

Sensor inercial

Para este proyecto se decidió utilizar sensores inerciales, ya que como se mencionó en la revisión de la literatura, son los que proporcionan información más relevante para clasificar actividades, además de que podemos encontrar modelos muy pequeños y de muy bajo consumo de energía, como lo es el sensor *MPU-9150* incluido en diseño del sistema. El MPU-9150 es una unidad de medición inercial, la cual cuenta con un acelerómetro, un giroscopio y un magnetómetro integrados en el mismo chip. Todos ellos pueden tomar muestras en los 3 ejes x, y y z. Para este proyecto solo se utilizaron los sensores acelerómetro y giroscopio ya que se consideró que el magnetómetro tiene un consumo de energía elevado y no proporciona información lo suficientemente útil comparado con el acelerómetro y el giroscopio.

El acelerómetro en el MPU-9150 mide la aceleración lineal en cada eje y el giroscopio la velocidad angular de cada eje. Este integrado se caracteriza por tener un empaquetado pequeño (4 mm x 4 mm) y por ser de muy bajo consumo de energía. Las principales características de los sensores acelerómetro y giroscopio se muestran en la siguiente lista.

- Acelerómetro:
  - Ejes x, y y z.
  - $\pm 16g$ de rango a escala completa.
  - ADC de 16 bits.
  - Detección de toques.
  - Detección de umbrales.
  - Detección de orientación.
  - Frecuencia de muestreo de hasta 1 KHz.
- Giroscopio:

- Ejes x, y y z.
- $\pm 2000°/sec$ de rango a escala completa.
- ADC de 16 bits.
- Filtro digital programable.
- Frecuencia de muestreo programable de hasta 8 KHz.

El MPU-9150 cuenta con una interfaz de comunicación I2C para transmitir los datos de los sensores al microcontrolador.

Memoria F-RAM

La memoria F-RAM FM24VN10 del fabricante *Cypress* es una memoria no volátil que se caracteriza por tener una velocidad de escritura muy rápida a diferencia de otras memorias no volátiles como las EEPROM y las memorias flash. El microcontrolador no necesita esperar a que se escriban los datos en la memoria ya que la escritura ocurre de manera casi instantánea. La memoria utiliza el protocolo I2C para comunicarse con el microcontrolador a una velocidad de hasta 3.4 MHz. Además, consume solamente 175 μA mientras escribe y 5 μA mientras está en reposo, lo que la hace ideal para nuestra aplicación. Esta memoria cuenta con 1 Mbit de almacenamiento, suficiente para almacenar aproximadamente media hora de muestras de los sensores. La memoria F-RAM servirá para almacenar en formato binario los datos que se obtienen de los sensores durante la etapa de recolección de muestras para el armado de la base de datos.

Memoria Micro SD

Para transferir los datos obtenidos de los sensores a la computadora, se añadió al sistema una ranura para tarjetas Micro SD. Aprovechando la gran capacidad de almacenamiento de este tipo de memorias, se pueden recolectar horas de muestras de los sensores. Además, las bibliotecas existentes para controlar memorias micro SD para el microcontrolador MSP430 facilitan la programación del almacenamiento de los datos en ésta. Al finalizar un ejercicio de recolección, en la memoria Micro SD se almacenará en un archivo de texto los datos provenientes de los sensores que se guardaron primero en formato binario en la memoria F-RAM. La memoria puede ser extraída fácilmente para transferir los datos recabados a una computadora.

Botones y diodos LED

Para la interacción con el usuario de una manera simple durante la etapa de recolección de muestras, se añadieron al diseño botones y diodos LED. En total, se incluyeron 4 botones y 3 diodos LED. Dos de los botones se utilizan para indicarle al dispositivo a qué actividad pertenecen los datos que se van a recolectar para posteriormente indicar en el archivo de texto almacenado en la Micro SD el número de actividad a la que corresponden los datos. Un tercer botón sirve para iniciar o suspender la recolección de datos, y el último botón se utiliza para reiniciar el funcionamiento de la tarjeta. Los diodos LED indican al usuario la actividad seleccionada en formato binario, es decir, un LED encendido indicará un uno, y apagado indicará cero. También se utilizan para indicar cuándo se debe comenzar a realizar la actividad seleccionada y cuándo se ha terminado la recolección de muestras.

Alimentación

Para alimentar al dispositivo se utiliza una batería de polímero de litio o Li-Po. Este tipo de baterías se caracterizan por ser ligeras, pueden fabricarse con casi cualquier forma o tamaño y son recargables. Esto las hace ideales para dispositivos tipo *wearables*. La batería seleccionada tiene 420 mAh de capacidad y unas dimensiones de 4cm x 3cm x 0.4cm. En el diseño de este dispositivo no se incluyen los componentes necesarios para la carga de la batería, por lo que esta debe recargarse con un cargador externo. En el segundo diseño ya se incluye este circuito y un puerto micro USB para la carga.

4.1. Diagrama esquemático del dispositivo

En la figura 4.3 podemos ver un diagrama ilustrativo del diseño del dispositivo en el que se puede apreciar cómo se conectan entre sí los módulos descritos anteriormente, los voltajes de alimentación y protocolos de comunicación utilizados.

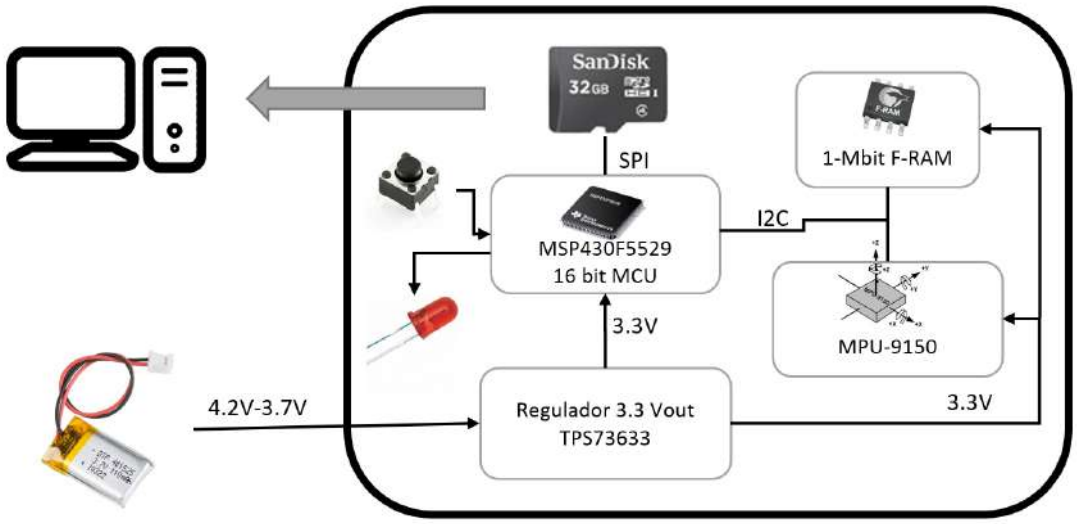

Figura 4.3: Diagrama del diseño del dispositivo para el reconocimiento de actividades físicas.

En el diseño esquemático del dispositivo se muestran los diferentes componentes electrónicos del sistema, las conexiones eléctricas correspondientes a señales y a líneas de alimentación del sistema. En las figuras 4.4 a 4.9 se encuentra el diagrama esquemático del dispositivo diseñado en este proyecto. A continuación se describen de manera breve los circuitos que se encuentran en las imágenes.

En la figura 4.4 se encuentra el conector de la batería Li-Po (**JP1**). Debido a que el voltaje de este tipo de baterías varía de 4.2 a 3.7 volts según el nivel de carga que tengan, se añadió un regulador (**U8**) **TPS73633** fijo de 3.3 Volts de salida que solamente necesita de capacitores de desacoplo como componentes externos. El voltaje de salida de 3.3V se utiliza para alimentar el microcontrolador, las memorias F-RAM y micro SD, los diodos LED y el sensor inercial.

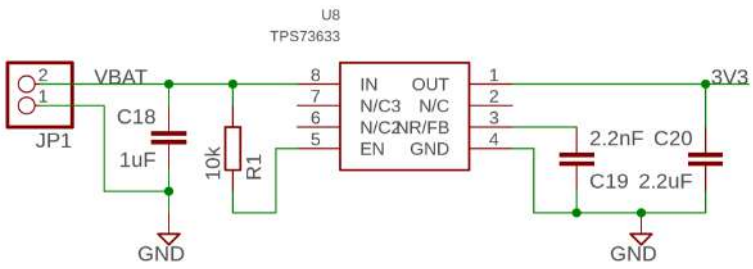

Figura 4.4: Diseño esquemático (1/6).

En la figura 4.5 se encuentra el diagrama de conexión del microcontrolador (**U2**) , del conector de programación y el de comunicación con la PC que es a

través de comunicación serial UART (**SV1**). Para programar el microcontrolador se utiliza la tarjeta de desarrollo MSP-EXP430F5529LP la cual integra un programador que utiliza el protocolo *2-Wire JTAG*. En la figura 4.6 se encuentran los botones y los diodos LED para el control del dispositivo por parte del usuario y en la figura 4.7 se encuentra la memoria F-RAM **FM24VN10** la cual se comunica con el microcontrolador a través de comunicación serial I2C. Este solamente necesita un capacitor de desacoplo y 2 resistores para la comunicación serial.

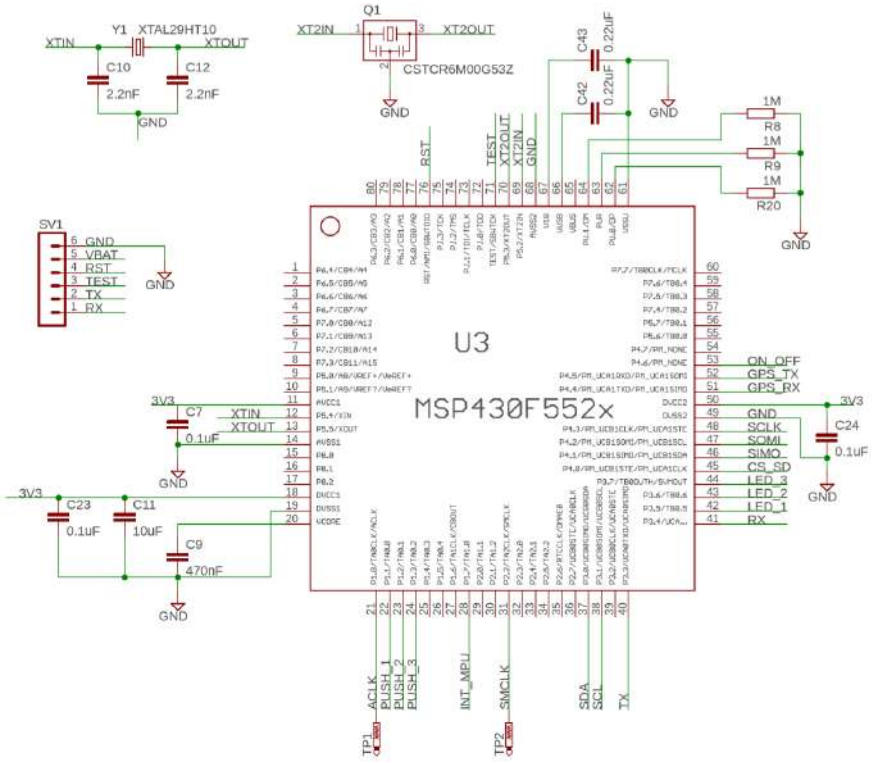

Figura 4.5: Diseño esquemático (2/6).

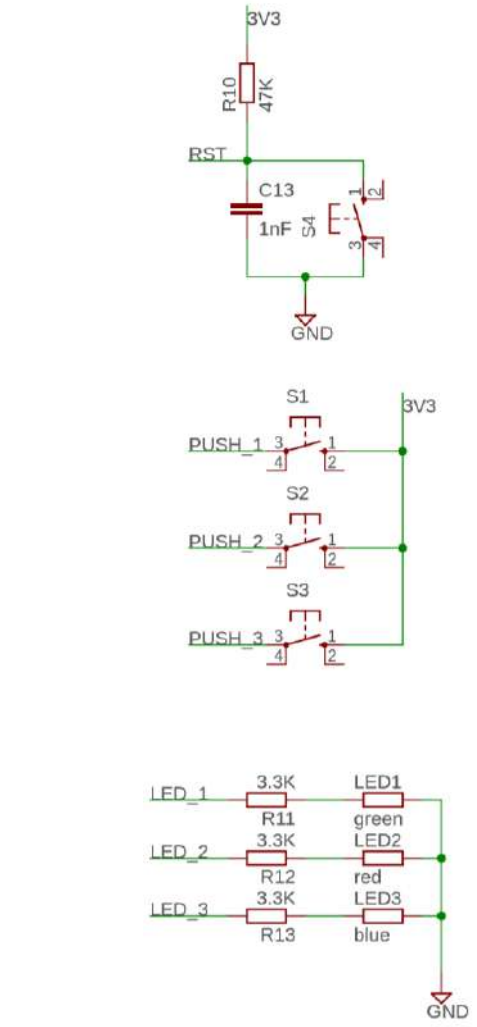

Figura 4.6: Diseño esquemático (3/6).

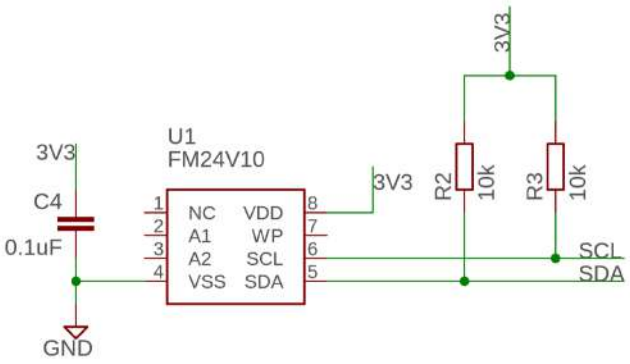

Figura 4.7: Diseño esquemático (4/6).

En la figura 4.8 se encuentra la unidad de medición inercial. Se comunica con el microcontrolador utilizando comunicación I2C y una señal de interrupción

(**INT_MPU**) para indicar cuando se ha recolectado una nueva muestra. Como componentes externos solo necesita capacitores de desacoplo. Por último, en la figura 4.9 está el circuito del conector para memorias micro SD. Este se comunica utilizando comunicación serial SPI de 4 señales.

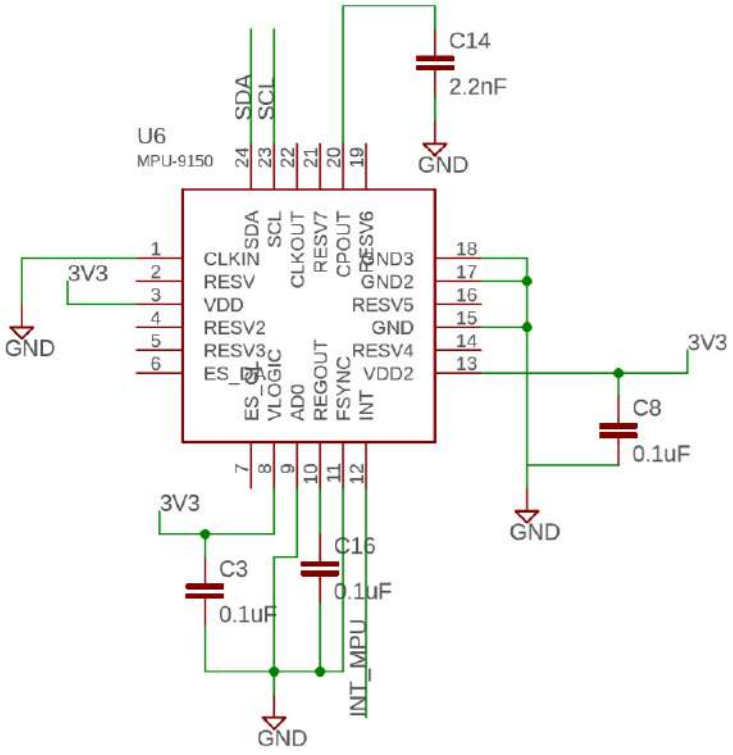

Figura 4.8: Diseño esquemático (5/6).

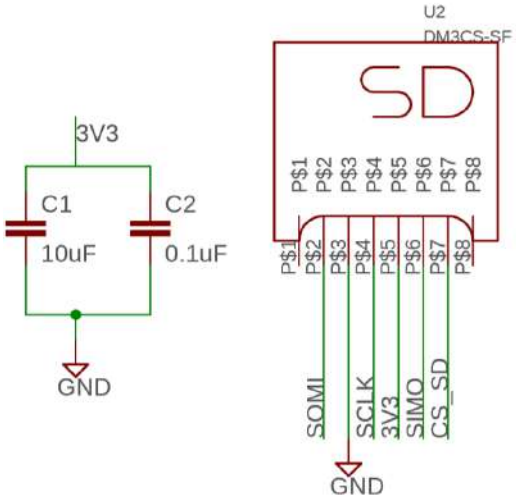

Figura 4.9: Diseño esquemático (6/6).

4.2. Fabricación del dispositivo

El *Layout* de la tarjeta de circuito impreso (PCB) se realizó en el software Cadsoft Eagle. En la figura 4.10 se encuentra el *Layout* de la cara superior y en la figura 4.11 el de la inferior. La tarjeta se fabricó utilizando el método de película fotosensible y posteriormente se realizó el montaje de los componentes. Las dimensiones de la tarjeta fabricada son 7.5 cm x 3.8 cm. En las figuras 4.12, 4.13 y 4.14 se encuentran imágenes del dispositivo terminado. Fueron realizadas varias pruebas para asegurar el funcionamiento correcto de todos los módulos que integran el dispositivo. Como se mencionó anteriormente, el dispositivo fue diseñado para facilitar los ejercicios de recolección de muestras de los sensores y su transferencia a la computadora para su posterior análisis, y además los componentes se eligieron tomando en cuenta las restricciones existentes debido al método de fabricación. Por este motivo, el tamaño, forma y consumo de energía del dispositivo no es el óptimo para un dispositivo *wearable*. Posteriormente se presenta el diseño de un segundo dispositivo de menor tamaño y consumo de energía, realizado con base a los resultados de las pruebas sobre el primero.

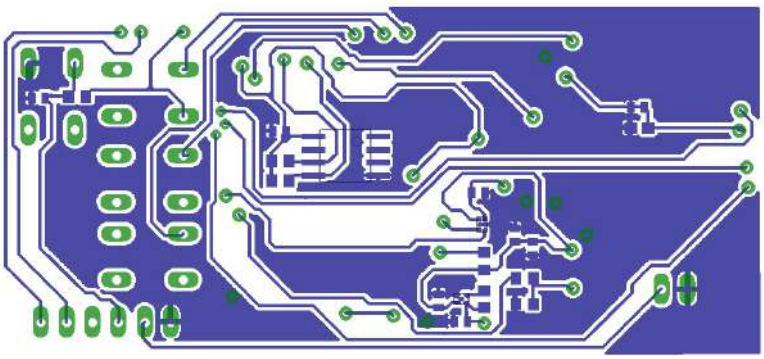

Figura 4.10: Vista superior del diseño de la tarjeta de circuito impreso.

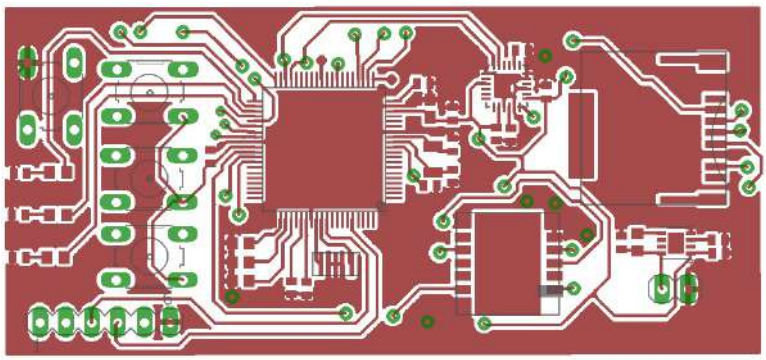

Figura 4.11: Vista inferior del diseño de la tarjeta de circuito impreso.

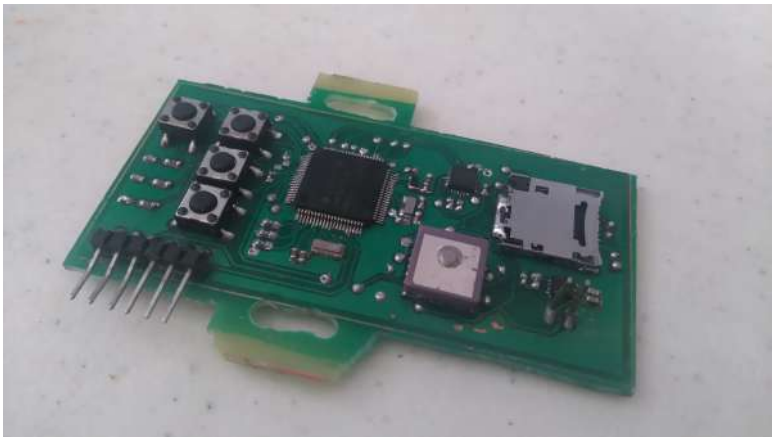

Figura 4.12: Foto del dispositivo de detección de actividades fabricado (1/3).

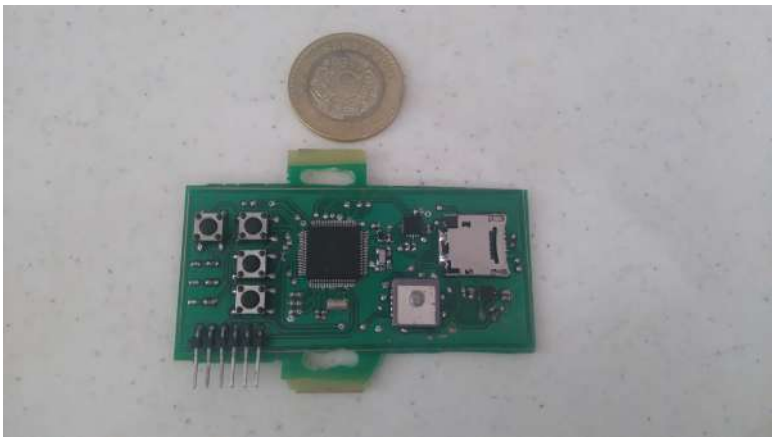

Figura 4.13: Foto del dispositivo de detección de actividades fabricado (2/3).

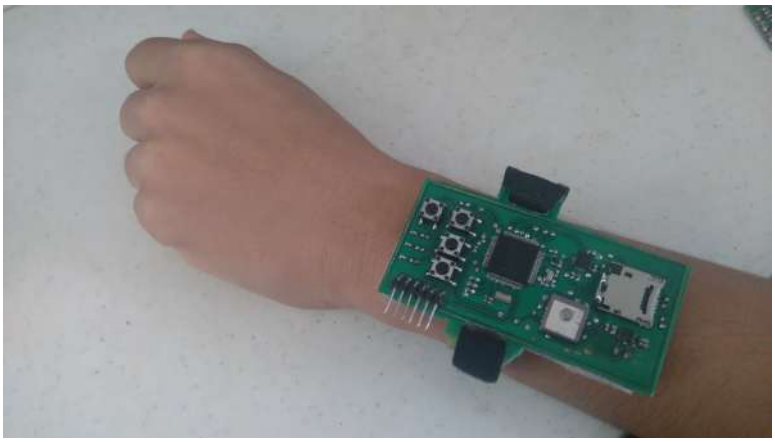

Figura 4.14: Foto del dispositivo de detección de actividades fabricado (3/3).

Capítulo 5

Diseño y pruebas del sistema de clasificación actividades físicas

5.1. Recolección de muestras y construcción de base de datos de actividades

Una vez teniendo el dispositivo, el primer paso para el desarrollo del sistema de reconocimiento de actividades físicas es construir una base de datos que contenga muestras de todos los sensores con la respectiva actividad a la que corresponde, es decir, muestras etiquetadas. Esta base datos servirá posteriormente para el entrenamiento de los algoritmos clasificadores de aprendizaje supervisado.

Como nuestro objetivo es detectar actividad física se decidió recolectar datos de algunas actividades físicas más comunes que la gente realiza en sus rutinas de ejercicios. En total fueron 6 actividades contando el estado de reposo:

1. Estar en reposo.
2. Caminar balanceando los brazos.
3. Correr balanceando los brazos.
4. Realizar abdominales.
5. Realizar lagartijas.
6. Realizar saltos verticales.

Para mejorar la flexibilidad del sistema de clasificación de actividades físicas es importante tener la mayor cantidad de muestras en la base de datos y, además, que estas sean de diferentes personas realizando las diferentes actividades. Esto es debido a que existen varias formas diferentes de realizar cada una

de las actividades. De esta forma, el clasificador entrenado podrá reconocer de mejor manera actividades de personas cuyos datos no hayan sido utilizados para el entrenamiento del mismo.

Primero, se desarrollo el firmware del microcontrolador. Para ello, se utilizó el IDE de *Texas Instruments*, Code Composer Estudio, en el que se codificó el firmware en lenguaje C. El programa recoge muestras constantemente de los sensores mientras se realiza una actividad, se guardan en formato binario en la memoria F-RAM y al terminar la recolección, el microcontrolador crea un archivo de texto en la memoria Micro SD en donde se escriben los valores recolectados en formato de texto. El funcionamiento detallado del dispositivo para realizar la recolección de datos de sensores mientras se realizan las actividades enlistadas anteriormente se describe a continuación.

Los leds y los botones en el dispositivo se utilizan para indicarle al usuario la actividad de la cual se desea obtener muestras de aceleración y velocidad angular. Para ello se le asignó un número a cada una de las actividades a entrenar. Los 3 leds indican un número en binario del 000 al 111 en donde un led encendido significa un uno y apagado un cero. La siguiente lista muestra el número que se le asignó a cada actividad.

- Reposo - 0
- Camiar - 1
- Correr - 2
- Realizar abdominales - 3
- Realizar lagartijas - 4
- Realizar saltos verticales - 5

Con dos de los botones se selecciona el número de actividad de la cual se recolectarán datos. Uno de ellos se utiliza para incrementar la cuenta y el otro para decrementarla. Los diodos LED desplegarán en código binario el número correspondiente. Al presionar el tercer botón los 3 LED se apagarán y comenzarán a encender de uno en uno cada segundo indicando y dándole tiempo al usuario de comenzar a realizar la actividad seleccionada antes de que el dispositivo comience a almacenar muestras. Cuando este tiempo pase, el dispositivo comienza a leer muestras de aceleración y velocidad angular del sensor inercial a una frecuencia de 10 muestras cada segundo y las va almacenando en la memoria F-RAM en

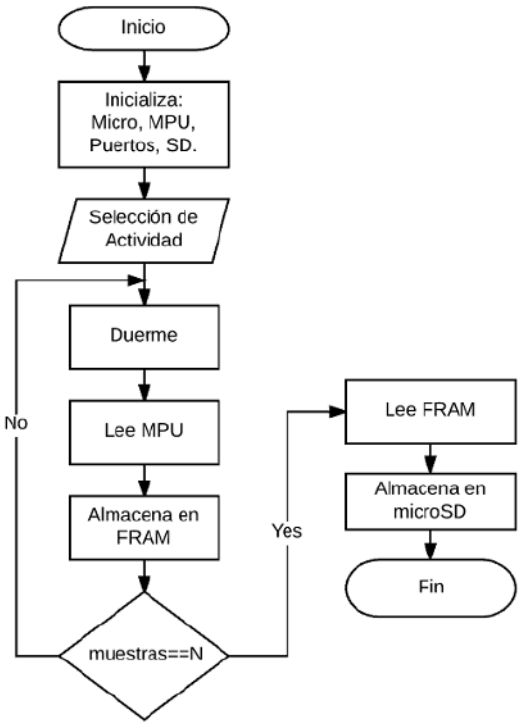

Figura 5.1: Diagrama de flujo del funcionamiento del dispositvo para la recolección de muestras de aceleración y velocidad angular.

formato binario. Mientras esto sucede, el LED azul parpadea indicando que el usuario debe continuar realizando la actividad. Esto ocurre durante 10 segundos, haciendo un total de 100 muestras. Al finalizar el muestreo, los 3 diodos LED parpadearán indicando que se ha terminado la recolección. En este momento, si se considera que los datos recabados son útiles para el entrenamiento , es decir, que el usuario realizó la actividad seleccionada de manera natural y durante los 10 segundos, entonces se presiona el tercer botón para indicarle al dispositivo que debe almacenar la muestras recolectadas guardadas en la F-RAM, en la memoria micro SD en un archivo de texto. Cada una de las filas de este archivo contendrá 7 valores: la aceleración en los 3 ejes, la velocidad angular en los 3 ejes y la etiqueta que será el número correspondiente a la actividad. De esta manera se realiza la recolección de muestras de entrenamiento de una actividad de una persona. El mismo proceso se repite para cada actividad y con diferentes personas. El diagrama de flujo del programa en el microcontrolador para la recolección de muestras de los sensores se encuentra en la figura 5.1

Se le pidió a 9 voluntarios que se colocaran en la muñeca izquierda el dispositivo y realizaran las 6 actividades. Por cada persona se recolectaron 10 segundos de muestras de aceleración y velocidad angular de cada actividad, en la manera

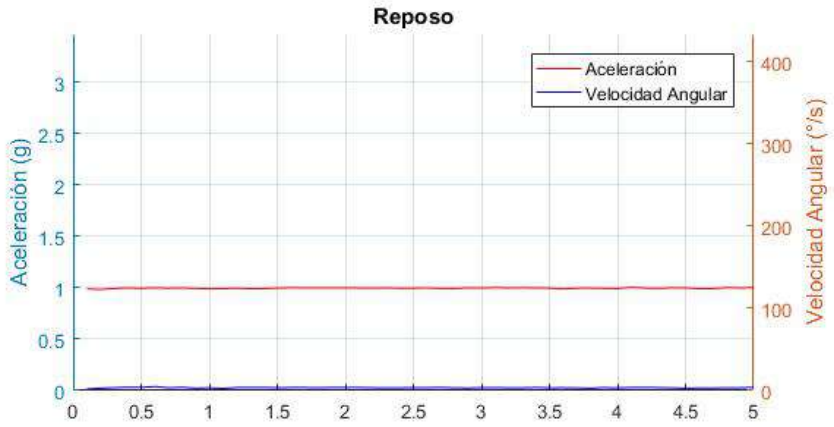

Figura 5.2: Gráfica de las señales de aceleración y velocidad angular de la actividad de reposo.

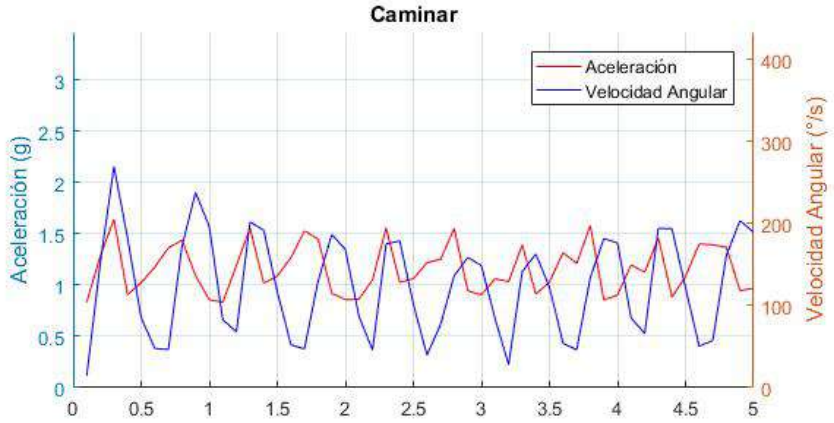

Figura 5.3: Gráfica de las señales de aceleración y velocidad angular de la actividad de caminar.

en que se explicó anteriormente. En las gráficas de las figuras 5.2 a5.7 se pueden apreciar las señales de aceleración y velocidad angular en el dominio del tiempo de los diferentes ejercicios.

A simple vista, se puede notar la diferencia en las gráficas de las señales de un ejercicio a otro, lo que hace pensar que la clasificación pueda lograrse utilizando características en el dominio del tiempo, evitando así los costosos cálculos necesarios para la transformación a dominio de frecuencia que pudieran reducir la autonomía del dispositivo.

5.2. Extracción de características

Como se mencionó en el marco teórico, un algoritmo clasificador recibe como entrada un vector cuyos elementos describen lo que se desea clasificar llamado

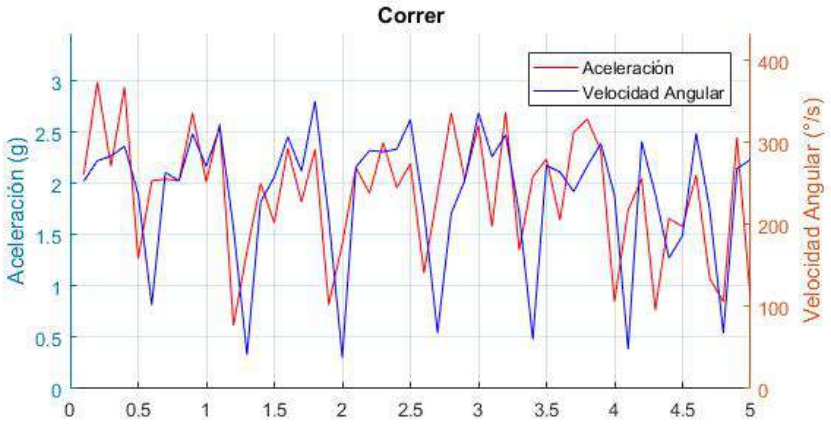

Figura 5.4: Gráfica de las señales de aceleración y velocidad angular de la actividad de correr.

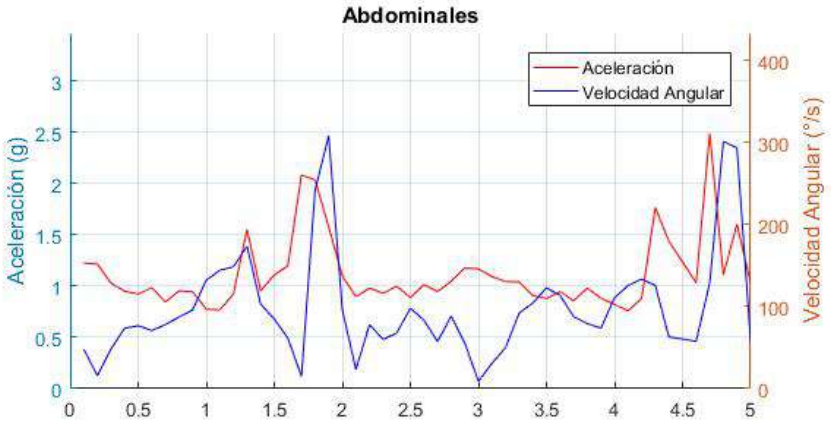

Figura 5.5: Gráfica de las señales de aceleración y velocidad angular de la actividad de realizar abdominales.

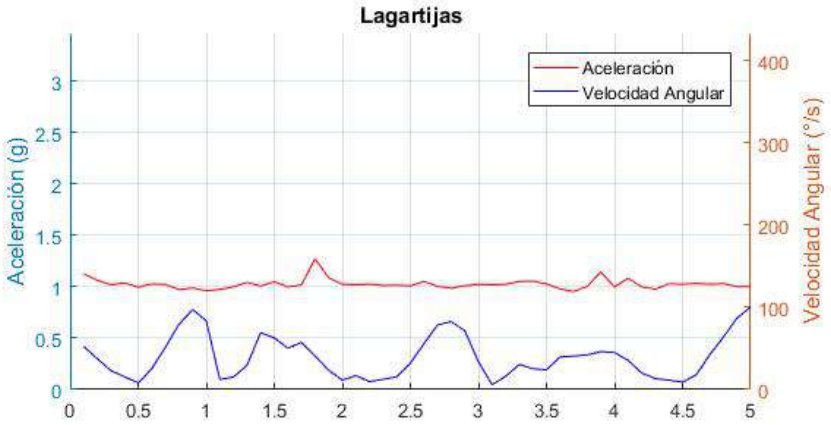

Figura 5.6: Gráfica de las señales de aceleración y velocidad angular de la actividad de realizar lagartijas.

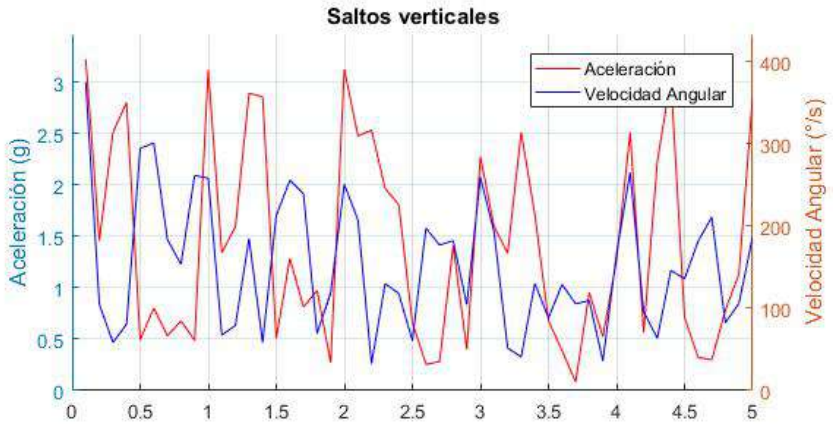

Figura 5.7: Gráfica de las señales de aceleración y velocidad angular de la actividad de realzar saltos verticales.

vector de características. En nuestro caso, las características encapsularán la información de las señales de aceleración y velocidad de angular.

La selección de características generalmente se realiza de manera empírica y estas deben contener información que facilite al clasificador discriminar entre las diferentes clases.

Como se desea que el clasificador funcione en tiempo real en un dispositivo de bajos recursos y que además sea de bajo consumo de energía, las características deben seleccionarse teniendo en cuenta la complejidad de su cálculo. Por esta razón, se decidió utilizar características simples que sean relativamente fáciles de calcular.

Los valores de aceleración y velocidad angular que se obtiene de los sensores sólo son válidas para el instante de tiempo en el que se midieron. Balancear el brazo al caminar o correr, hacer un abdominal, lagartija o dar un salto son movimientos que pueden durar hasta varios segundos, por lo que la aceleración y velocidad angular de la muñeca medidas en un instante de tiempo proporcionarían muy poca información del movimiento entero. Por esta razón, para el calculo de un vector de características se debe tomar en cuenta una serie de muestras consecutivas que recolecten información del movimiento completo que se desea clasificar. Al número de muestras consecutivas que se tomarán en cuenta para el cálculo de las características lo denotaremos como N.

Por cada muestra de datos, excepto para las primeras $N-1$ muestras capturadas, se calculará un vector de características tomando en cuenta la muestra actual y las $N-1$ muestras anteriores. A esto se le conoce como ventana deslizante y consiste en que dentro de una ventana estarán los últimos N valores capturados y al llegar una nueva muestra, la más antigua queda fuera de la ventana para

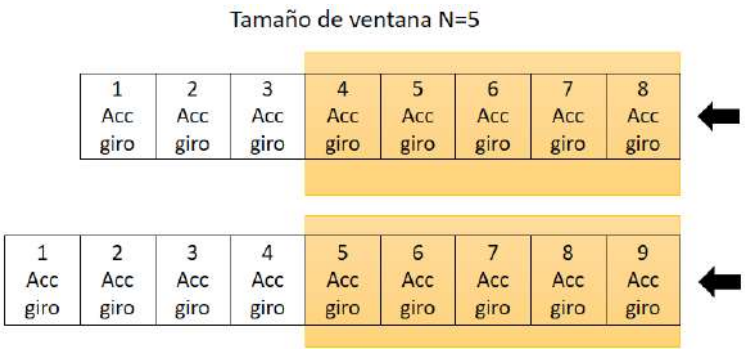

Figura 5.8: Ejemplo de ventana deslizante.

dar lugar a la nueva. El vector de características se calculará sobre los valores en la ventana. En la figura 5.8 se encuentra un ejemplo de este concepto para un tamaño de ventana de $N = 5$.

En este proyecto se probaron diferentes combinaciones de características con el fin de encontrar el vector con el que se obtengan los mejores resultados en la clasificación de actividades. En la siguiente lista se encuentran las diferentes características con las que se realizaron pruebas, las primeras 8 en la lista se calculan sobre la muestra más reciente en la ventana y las demás sobre la magnitud del vector de aceleración y del vector de velocidad angular de todas las muestras en la ventana:

- Aceleración en el eje x.
- Aceleración en el eje y.
- Aceleración en el eje z.
- Velocidad angular en el eje x.
- Velocidad angular en el eje y.
- Velocidad angular en el eje z.
- Magnitud del vector de aceleración.
- Magnitud del vector de la velocidad angular.

  Las siguientes características se calcularon sobre las N muestras consecutivas de la magnitud de aceleración y de la velocidad angular.

- Máximo de la magnitud de aceleración.
- Máximo de la magnitud de la velocidad angular.

- Mínimo de la magnitud de aceleración.
- Mínimo de la magnitud de la velocidad angular.
- Media de la magnitud de aceleración.
- Media de la magnitud de la velocidad angular.
- Varianza de la magnitud de aceleración.
- Varianza de la magnitud de la velocidad angular.

La magnitud del vector de aceleración o velocidad angular se calcula utilizando la norma del vector:

$$|v| = \sqrt{x^2 + y^2 + z^2} \tag{5.1}$$

Diferentes vectores de características construidas con combinaciones de las características enlistadas y variando el tamaño de la ventana se utilizaron para probar los algoritmos clasificadores de aprendizaje supervisado. Como se puede notar, solo se utilizaron características simples en el dominio del tiempo.

5.3. Diseño del algoritmo de clasificación de actividades

En este proyecto se probaron dos técnicas de aprendizaje supervisado para el diseño del clasificador: K-NN y Redes Neuronales Artificiales. Estos métodos fueron descritos brevemente en el marco teórico.

Para construir la base de datos de actividades se le pidió a un total de nueve voluntarios que realizaran las 6 actividades mientras se recolectaban muestras en la forma en la que se describió en las secciones anteriores. Utilizando la misma base de datos, los métodos fueron probados siguiendo la metodología siguiente:

- Se define las características que se utilizarán para entrenar el algoritmo.
- Se define el tamaño de la ventana N.
- Se construye un conjunto de entrenamiento extrayendo vectores de características de la base de datos.
- Se entrena el algoritmo clasificador, se prueba y se mide el rendimiento.
- Se regresa al primer paso para probar con diferentes características, tamaños de ventana o parámetros del algoritmo.

- Se comparan los resultados obtenidos.

Para las pruebas, se programó un script en MATLAB que toma como parámetro el tamaño de ventana N, lee la base de datos, extrae los vectores de características etiquetados, normaliza estos vectores y los almacena en un archivo que posteriormente se utiliza como conjunto de entrenamiento para los algoritmos de aprendizaje. La normalización se utiliza para definir el rango de las características de 0 a 1 y evitar de esta manera que características con un amplio rango, como la varianza, tengan mayor peso en la clasificación que las características con rango más corto.

5.3.1. K-NN

El clasificador K-NN es un algoritmo simple y muy fácil de implementar. Se decidió probar este método para darse una idea de qué tan útil resultan los datos recolectados para realizar la clasificación de las actividades físicas enlistadas anteriormente, antes de implementar un algoritmo más complejo y que pueda ejecutarse en un dispositivo de bajos recursos.

El algoritmo K-NN no cuenta propiamente con una etapa de entrenamiento previa en la que la información adquirida se encapsula en ciertos parámetros. En vez de eso, cada que K-NN va a realizar la clasificación de una muestra, consulta el conjunto de entrenamiento para encontrar la clase a la que pertenece esta muestra, por lo tanto, el conjunto de entrenamiento debe estar siempre disponible. Para medir el rendimiento de este método se probaron distintos conjuntos de entrenamiento construidos utilizando diferentes tamaños de ventana. Además, también se probaron distintos valores del parámetro K.

El algoritmo K-NN fue implementado en un script de MATLAB. Este recibe como parámetro el valor de K (número de vecinos más cercanos), toma cada una de las muestras en el conjunto de entrenamiento y las clasifica utilizando todas las demás, y por último, compara las etiquetas con los resultados de la clasificación para calcular el porcentaje de acierto.

Para la primera prueba se formó un conjunto de entrenamiento tomando un tamaño de ventana de 10 muestras (un segundo) para la extracción de las características. Para el vector de características, se utilizaron todos los valores enlistados en la sección anterior. Con el script de MATLAB se probó K-NN con K de 1 a 10. El rendimiento se midió dividiendo la cantidad de ocasiones en las que el algoritmo clasificó una muestra correctamente entre el número de clasificaciones que realizó. La gráfica con los resultados se encuentra en la figura 5.9. En esta

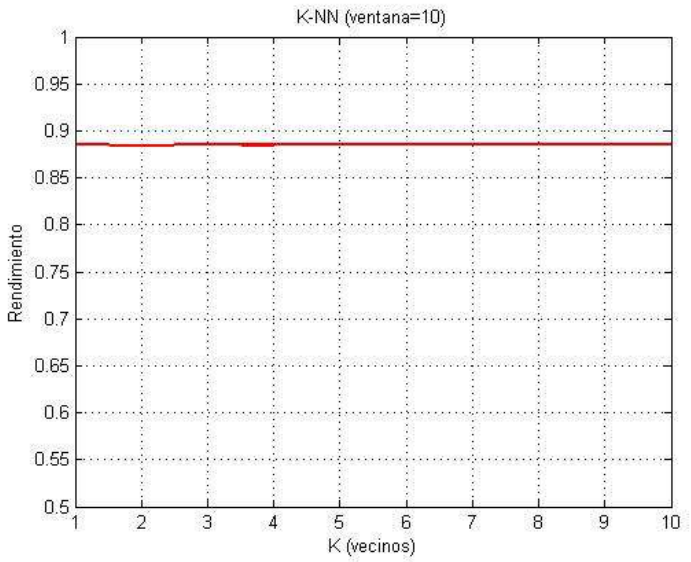

Figura 5.9: Rendimiento del algoritmo K-NN utilizando una ventana de tamaño 10.

gráfica podemos observar que no existe una diferencia notable en el rendimiento del algoritmo clasificador al incrementar el número K de vecinos.

Debido a que cualquier valor de K arroja resultados similares (cerca de 88 % de acierto), se seleccionó $K = 1$ para la siguiente prueba. Se probó K-NN con $K = 1$ utilizando diferentes conjuntos de entrenamiento que se construyeron variando el tamaño de ventana. Esto con el fin de ver como afecta este parámetro en la clasificación de las actividades. En la figura 5.10 podemos ver la gráfica con los resultados de esta prueba. Se puede observar como el tamaño de ventana de ventana afecta directamente el rendimiento del algoritmo. Esto es debido a que, como se mencionó anteriormente, mientras más grande sea la ventana, más información acerca del movimiento se encapsulará en las características. Como podemos ver, a partir de una ventana de 2 segundos se logra un rendimiento mayor al 95 % ya que generalmente los movimientos que se realizan con el brazo para realizar una repetición de alguna de las actividades físicas no duran más de dos segundos.

Ya que mientras más grande sea la ventana, la complejidad en tiempo del cálculo de las características aumenta, es importante que al escoger el tamaño de ventana se realice un balance entre el rendimiento del algoritmo y la complejidad o consumo de recursos del mismo.

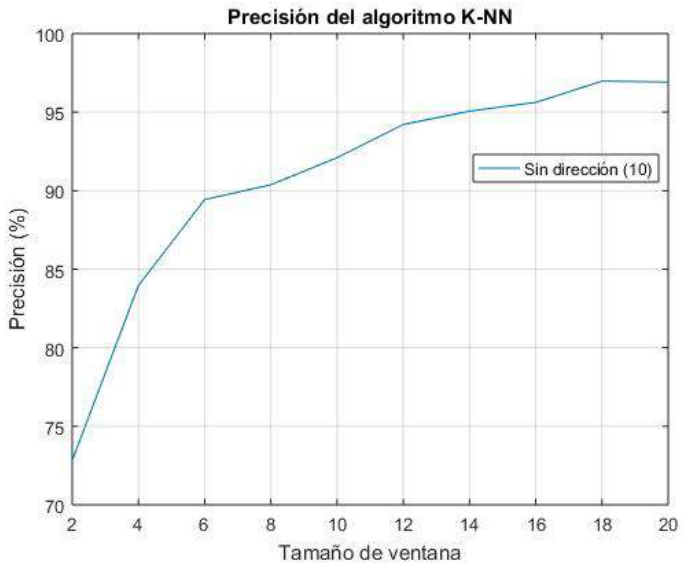

Figura 5.10: Rendimiento del algoritmo K-NN con $K = 1$ para diferentes tamaños de ventana.

5.3.2. Redes Neuronales

El siguiente método que se probó fue el de Redes Neuronales Artificiales. Para ello, se codificó en lenguaje C++ un programa que ejecuta el algoritmo de entrenamiento de retro-propagación, el cual fue descrito en el marco teórico. El programa también realiza la evaluación del clasificador resultante del entrenamiento. Este se ejecuta de manera *offline* en una computadora utilizando los datos recolectados para el entrenamiento y también para la evaluación.

El programa utiliza como parámetro la topología de la red, es decir, el número de capas ocultas y cuantas neuronas tiene cada capa. Este utiliza la validación cruzada para realizar el entrenamiento y prueba del clasificador. La validación cruzada es un técnica utilizada para realizar una evaluación del rendimiento del algoritmo más realista en la que se demuestra el rendimiento al clasificar nuevos datos y no los datos con los que fue entrenado.

Como función de activación de las neuronas de la red se utilizó la tangente hiperbólica, ya que su aproximación puede implementarse en un microcontrolador y ejecutarse en menor tiempo que la función sigmoide. La aproximación a la que llamamos *fast-tanh* está dada por la siguiente ecuación:

$$fast\_tanh(x) = \frac{(((x^2 + 378)x^2 + 17325)x^2 + 135135)x}{((25x^2 + 3150)x^2 + 62370)x^2 + 135135} \tag{5.2}$$

Con esta función se obtiene una buena aproximación a la función tangente hiperbólica pero solo en el intervalo $-5 < x < 5$ como podemos notar en la gráfica

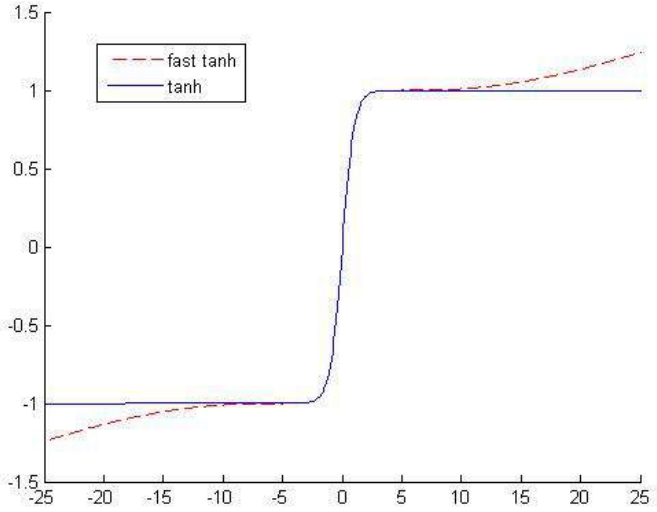

Figura 5.11: En color azul se encuentra la gráfica de la función tangente hiperbólica y en color rojo la aproximación a esta función .

de la figura 2.3. Sin embargo, también podemos observar que $tanh(x) \cong -1$ para $x < -5$ y $tanh(x) \cong 1$ para $x > 5$, por lo que simplemente si $tanh(x) < -1$ o $tanh(x) > 1$ se limita la salida a -1 y 1 respectivamente.

El funcionamiento del programa para el entrenamiento de la Red Neuronal se detalla a continuación. Primero, el programa revuelve al azar las muestras de entrenamiento, después divide la base de datos en 5 particiones de igual número de muestras, y dentro de cada partición el número de muestras de cada clase también es el mismo. Se ejecuta el algoritmo de entrenamiento utilizando 4 de las particiones como conjunto de entrenamiento y se evaluá el rendimiento del clasificador resultante utilizando la quinta partición como prueba. Se realiza el mismo proceso tomando otra partición como prueba y el resto como entrenamiento. Este proceso se repite hasta que se hayan usado las 5 particiones como prueba y se promedian los rendimientos obtenidos de cada iteración. Este último valor nos indica qué tan buena es la topología seleccionada o el tamaño de ventana que se utilizó para construir el conjunto de entrenamiento.

Se realizaron pruebas con distintas topologías de red para encontrar con cual se obtienen los mejores resultados en la clasificación de actividades. Para esta prueba, se utilizó un conjunto de entrenamiento formado tomando una ventana de tamaño 1 segundo. En la tabla 5.1 se comparan los resultados obtenidos para topologías de una sola capa oculta. En la gráfica de la figura 5.12 podemos apreciar que a mayor número de neuronas aumenta el rendimiento, pero este deja de aumentar después de 5 neuronas. En la tabla 5.2 se encuentran los resultados

Cuadro 5.1: Rendimiento obtenido utilizando una capa oculta.

Neuronas en la capa oculta	2	3	4	5	6	7	8	9	10	15	20
Rendimiento	81 %	86 %	89 %	91 %	91 %	92 %	91 %	92 %	93 %	92 %	93 %

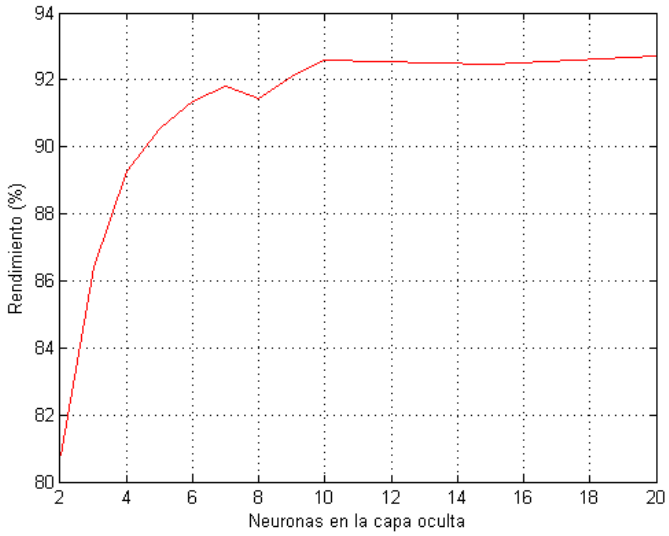

Figura 5.12: Rendimiento obtenido en topologías de una sola capa oculta.

de las topologías con más de una capa oculta. En este caso podemos ver que el aumentar el número de capas no aumenta el rendimiento.

Para la siguiente prueba se decidió utilizar la topología de una capa oculta con 5 neuronas ya que se consideró que se logra una buena precisión de clasificación (91 %) con pocas neuronas. Esta prueba consistió en ir incrementando el tamaño de la ventana para construir el conjunto de entrenamiento, entrenar la Red Neuronal y medir su precisión. Los resultados de esta prueba se grafican en la figura 5.13. Podemos ver que con un segundo de ventana (10 muestras) se lográ un rendimiento mayor al 90 %. Con 5 segundos se alcanza casi el 100 % de preci-

Cuadro 5.2: Rendimiento obtenido con 2 y 3 capas ocultas.

Capa				
Oculta 3	Oculta 2	Oculta 1	Salida	Rendimiento
	2	2	6	69 %
	3	2	6	81 %
	5	2	6	79 %
	2	5	6	71 %
	5	5	6	90 %
	10	10	6	91 %
10	8	6	6	77 %

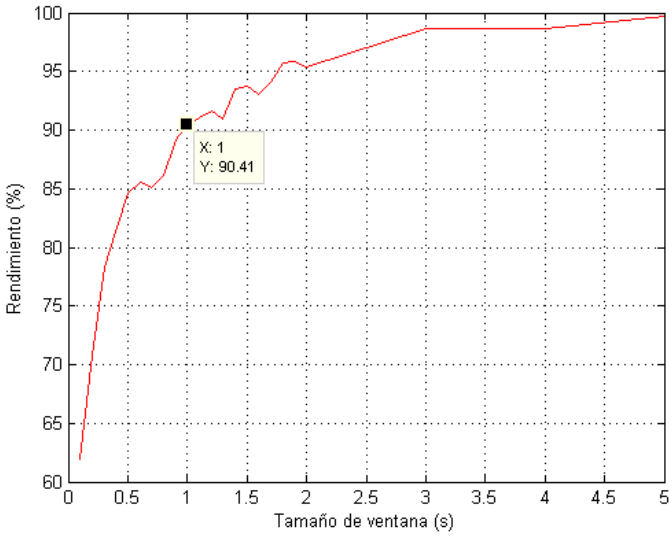

Figura 5.13: Rendimiento de la Red Neuronal con una capa oculta de 5 neuronas variando el tamaño de ventana.

		Resultado de la clasificación					
		Reposo	Caminar	Correr	Abdominales	Lagartijas	Saltos
Clase real	Reposo	99.62	0	0	0	0.37	0
	Caminar	0	91.32	0	7.16	0	1.5
	Correr	0	1.5	90.56	0.18	0	7.73
	Abdominales	0	13.58	4.71	71.13	5.66	4.9
	Lagartijas	0.18	0	0	1.13	98.67	0
	Saltos	0	0.18	6.98	0.18	0	92.64

Cuadro 5.3: Matriz de confusión de los resultados obtenidos con la Red Neuronal de una capa oculta con 5 neuronas utilizando un tamaño de ventana de 1 segundo (10 muestras).

sión, sin embargo la complejidad del cálculo de las características se incrementaría notablemente.

En la tabla 5.3.2 se detallan los resultados en una matriz de confusión, obtenidos por una Red Neuronal de una capa oculta con 5 neuronas entrenada utilizando una ventana de tamaño N=10 (1 segundo) para la extracción de características.

Hasta ahora, todos las pruebas que se han mostrado fueron realizadas de manera *offline* en una computadora, esto con el fin de encontrar el tamaño de ventana y la topología de red con la que obtengamos un buen balance entre precisión y complejidad computacional. En la siguiente sección se analizará la implementación del clasificador de actividades en el dispositivo descrito en secciones anteriores.

5.4. Implementación en tiempo real

Para la implementación en tiempo real del clasificador de actividades se decidió utilizar la Red Neuronal con una capa oculta de 5 neuronas. Utilizando el software IDE de Texas Instruments, Code Composer Studio, se codificaron los algoritmos de lectura de datos de los sensores, cálculo de características, propagación en la Red Neuronal y almacenamiento de los resultados.

En la figura 5.14 se encuentra el diagrama de flujo del programa. Primero el microcontrolador realiza las configuraciones iniciales y configura al sensor inercial MPU-9150 para obtener muestras del acelerómetro y del giroscopio a una frecuencia de 10 Hz. Después, este entra en modo de bajo consumo de energía y permanece así hasta que el sensor manda una señal de interrupción indicando que se ha recolectado una muestra. Se lee la muestra del sensor y esta se almacena en un vector que representa la ventana. En este vector se irán almacenando las muestras y cuando esté lleno, la nueva muestra sobrescribirá la más antigua en el vector de tal forma que en este se encuentren las últimas N muestras. Cada que llegue una nueva muestra esta se almacena en la ventana y se calculan las características sobre los datos de este vector. Después, el vector de características resultante se propaga en la Red Neuronal que entrega la clase a la que pertenece como salida. En la memoria micro SD se va almacenando el tiempo que se realiza cada ejercicio según los resultados que vaya arrojando la Red Neuronal.

En un sistema embebido de bajos recursos y que funciona con batería, es muy importante tomar en cuenta la complejidad de los algoritmos que ejecuta. El dispositivo utilizado en este proyecto cuenta con un microcontrolador MSP430 con características muy limitadas si se compara con una computadora, por lo que es de importancia realizar un análisis del tiempo de ejecución que le toma al microcontrolador ejecutar los algoritmos de clasificación. Al momento de codificar los algoritmos existen factores a tener en cuenta que afectan la eficiencia del mismo y por lo tanto el uso de recursos. Por ejemplo, debido a que el MSP430 no cuenta con un módulo para operaciones de punto flotante se debe tratar de minimizar las operaciones con este tipo de dato y utilizar precisión simple en vez de doble si esto no representa un problema para la aplicación. También se debe tratar de evitar operaciones como la división, la raíz cuadrada o funciones trigonométricas ya que son costosas debido a la arquitectura que se está utilizando. En cuanto a la red neuronal, el número de neuronas en la red, el tamaña de la ventana para el cálculo de características y la función de activación utilizada, son parámetros que afectan directamente en el tiempo de ejecución al clasificar una muestra.

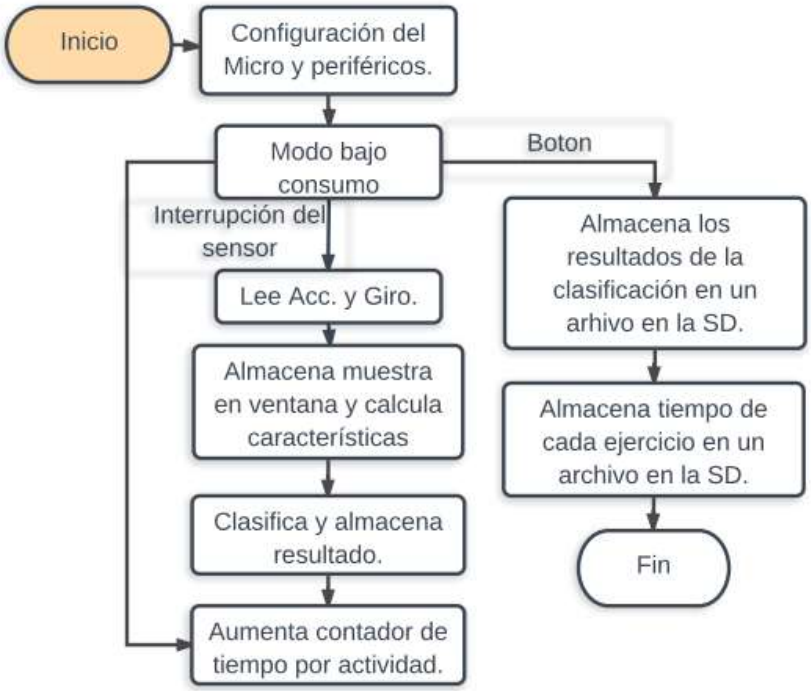

Figura 5.14: Diagrama de flujo del funcionamiento del dispositivo para el reconocimiento de actividades físicas.

Para medir como afectan estos parámetros a la eficiencia del algoritmo se realizaron varias pruebas con distintos escenarios, en donde se midió el tiempo que el microcontrolador se tarda en realizar el cálculo de características y el tiempo que se tarda propagando el vector en la Red Neuronal. Entre las pruebas está la utilización de variables de precisión doble o simple, utilizar la función de activación sigmoide o *fast-tanh*, utilizar distintos tamaños de ventana y omitir o no la raíz cuadrada en el cálculo de la magnitud de los vectores de aceleración y velocidad angular.

En la tabla 5.4 se encuentran los tiempos de ejecución medidos para los distintos escenarios. Primero, se puede observar que el utilizar precisión doble incrementa considerablemente el tiempo de ejecución. También podemos ver que utilizar la función de activación *fast-tanh* resulta en un tiempo de clasificación mucho menor y que al incrementar el tamaño de ventana se incrementa el tiempo de cálculo de características. Por último, omitiendo la raíz cuadrada en el cálculo de la magnitud el tiempo de ejecución mejora aunque no de manera muy notable.

Ya que la frecuencia de muestreo del sensor inercial es de 10 muestras por segundo, cada 100 milisegundos el microcontrolador deberá realizar el proceso de clasificación. Si se toma la combinación de utilizar precisión simple con una ven-

Cuadro 5.4: Comparación de tiempos de ejecución del algoritmo de clasificación de actividades en el microcontrolador MSP430F5529.

Precisión	Simple						Doble
F. Activación	Sigmoidal			Fast-Tanh			
Ventana	10	20	50	10	10	20	10
Raíz Cuadrada?	Si			Si	No		
Características(ms)	1.9	2.8	4.3	1.8	1.7	2.5	3.5
Clasificación(ms)	17.5	17.6	17.5	3.95	3.92	3.9	10.1
Total (ms)	19.4	20.4	21.8	5.75	5.62	6.4	13.6

tana de tamaño 10 y función de activación *fast-tanh* el microcontrolador tendría un ciclo de trabajo de apenas 5.62 %, sin embargo se debe tener en cuenta que la detección de actividades pudiera ser una función más de un dispositivo más complejo que pudiera incorporar un sensor de frecuencia cardíaca, contador de pasos, detección de sueño o una pantalla para interacción con el usuario.

Capítulo 6

Otra aplicación: Detección de caídas

Para este proyecto se habían seleccionado actividades relacionadas a ejercicios físicos, sin embargo, el mismo sistema desarrollado puede adaptarse fácilmente para detectar otro tipo de actividades.

En el área del cuidado de la salud, el cuidado del adulto mayor es un tema que ha estado adquiriendo interés en los últimos años. Según cifras de la OMS, las caídas son la segunda causa de muerte accidental a nivel mundial. En el caso de los adultos mayores es más el riesgo de sufrir lesiones graves al caerse y es importante que la persona sea atendida lo más pronto posible, por lo que un dispositivo que detecte la caída y además avise a un familiar o a emergencias podría evitar consecuencias graves.

Aprovechando el software y hardware hecho durante este proyecto, se decidió evaluar la implementación de un algoritmo de detección de caídas utilizando redes neuronales que pueda funcionar en el dispositivo *wearable*. Para el desarrollo del detector de caídas se siguió el mismo proceso que el del desarrollo del clasificador de actividades.

El clasificador en este caso cuenta con dos posibles salidas: caída y no caída. Su entrada siguen siendo características obtenidas de los datos del sensor inercial.

Para la etapa de recolección de muestras se utilizó el dispositivo cuyo diseño se describió anteriormente. Se le pidió que se pusieran el dispositivo en la muñeca izquierda y simularan los diferentes tipos de caídas que se enlistan a continuación:

- Caerse hacia el frente apoyándose con el brazo izquierdo.
- Caerse hacia el frente apoyándose con el brazo derecho.
- Caerse hacia atrás apoyándose con el brazo izquierdo.

- Caerse hacia atrás apoyándose con el brazo derecho.
- Caerse hacia atrás sin apoyarse con los brazos.

En total se recolectaron datos de 32 caídas. También se les pidió que realizaran actividades de la vida diaria que involucran movimientos con los brazos para la recolección de muestras negativas.

- Caminar.
- Sentarse
- Levantarse de estar sentado.
- Acostarse.
- Levantarse de estar acostado.
- Beber de un vaso.
- Llevar comida a la boca.
- Cepillarse los dientes.
- Agitar el brazo.
- Golpear una mesa con el brazo .

Para la etapa de extracción de características se decidió utilizar solamente la señal del acelerómetro con el fin de reducir el consumo de energía, ya que el giroscopio encendido consume aproximadamente 3.8 mA y la vida útil de la batería se reduce considerablemente, lo que para un dispositivo de detección de caídas que debe estar en constante funcionamiento sin que se deba realizar una recarga de la batería en varios días, no resultaría práctico. Ya que una caída puede ocurrir en una fracción de segundo se decidió utilizar una frecuencia de muestreo mayor para tener una mejor resolución de la señal al momento del impacto de la persona en el suelo. Se configuró el acelerómetro con 50 Hz de frecuencia de muestreo y con un rango de $\pm 4g$.

En la figura 6.1 podemos ver varias gráficas de la señal de la magnitud aceleración medida durante las caídas. Se puede observar en todas las gráficas que se produce un pico en la señal de aceleración que corresponde al impacto de la persona en el suelo.

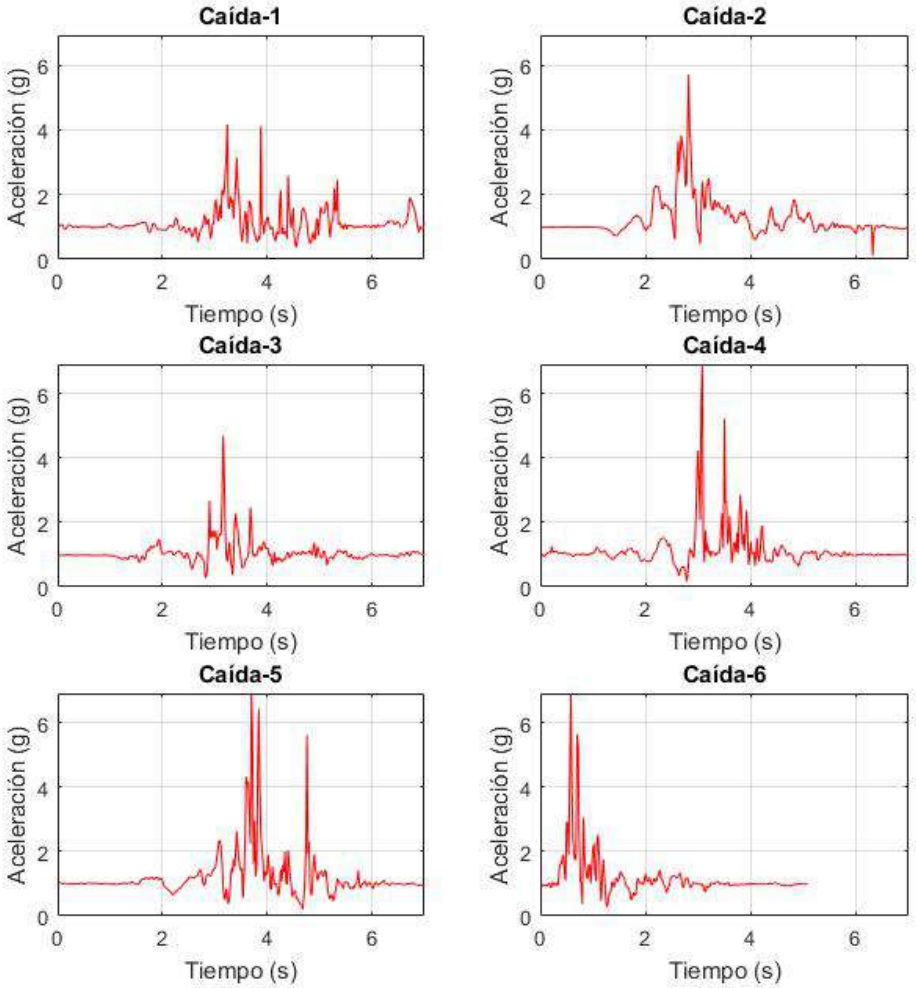

Figura 6.1: Gráficas de la señal de aceleración extraída de varias caídas.

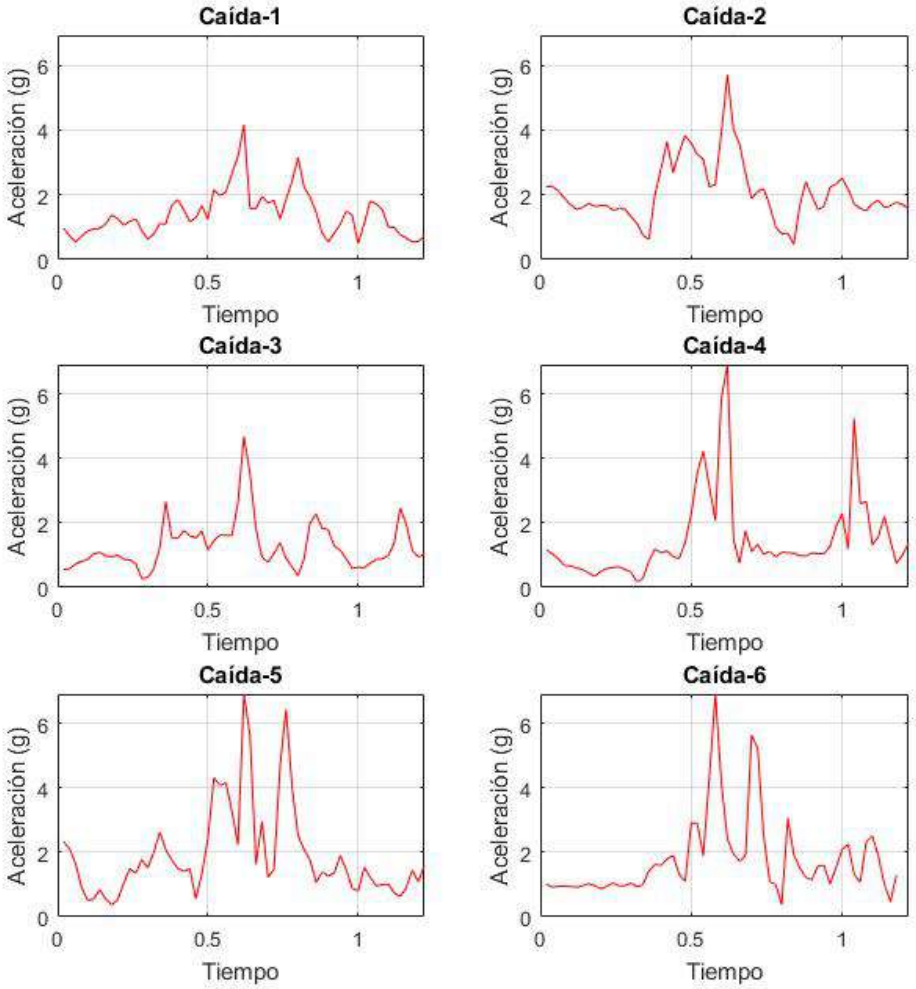

Figura 6.2: Grafícas de las señales de aceleración de caídas.

Como se puede ver, gran parte de las señales no corresponde a la señal de aceleración de una caída si no que son datos recolectados antes y después de que ocurriera la caída debido a la forma en la que se tomaron las muestras. Para solucionar esto, con un script de MATLAB se extrae una parte de las señales correspondiente a 30 muestras antes y 30 después del pico de la magnitud de aceleración, y sobre las señales resultantes (figura 6.2) se calcularán los vectores de características etiquetados como caídas.

De manera experimental, para la extracción de características se seleccionó una ventana de tamaño 40 muestras. De nueva cuenta se utiliza solamente la magnitud y no la dirección del vector para evitar que la posición en la que se coloque el dispositivo afecte la clasificación. Esto debido a que se recolectaron

pocas muestras de caídas. El vector de características seleccionado, también de manera experimiental, está conformado de los siguientes valores:

- La magnitud instantánea del vector de aceleración.
- La máxima magnitud en la ventana.
- La mínima magnitud en la ventana.
- La media de la señal de magnitud en la ventana.
- La varianza de la señal de magnitud en la ventana.
- La pendiente alrededor del máximo de magnitud dada por $2V_{imax} - V_{imax-1} - V_{imax+1}$ en donde V es el vector ventana e $imax$ es el índice del vector en donde se encuentra la máxima magnitud.
- La sumatoria del valor absoluto de las diferencias entre muestras consecutivas dado por: $\sum_{i=2}^{N} |V_i - V_{i-1}|$

Usando MATLAB se extrajeron los vectores de características de los archivos con las señales de aceleración y se creó el conjunto de entrenamiento. El mismo programa desarrollado con el algoritmo de retropropagación se utilizó para entrenar la Red Neuronal para reconocer caídas. De nuevo, se normalizaron las características antes del entrenamiento y se utilizó validación cruzada para entrenar la red y medir el rendimiento.

Utilizando una topología de una capa oculta con 5 neuronas se obtuvieron los resultados de la tabla 6.1. En ella podemos ver que el 97.2 % de las muestras pertenecientes a caídas fueron clasificadas correctamente, sin embargo, debido a que para cada señal de caída se extrajeron varios vectores de características, es muy probable que al menos una muestra de cada caída haya sido clasificada correctamente.

	Clasificado como caída	Clasificado como actividad
Muestras de caídas	97.2 %	2.8 %
Muestras de actividades	0 %	100 %

Cuadro 6.1: Matriz de confusión de los resultados obtenidos de las pruebas de la Red Neuronal para la detección de caídas.

Capítulo 7

Segunda versión del dispositivo

Como ya se ha mencionado, el dispositivo que se fabricó fue diseñado para realizar la recolección de muestras para el entrenamiento del clasificador de manera *offline* y posteriormente comprobar la eficiencia del clasificador funcionando en tiempo real en el dispositivo. Por este motivo, este incluía botones, diodos leds y una ranura para memorias micro SD para una fácil interacción con el usuario y posterior transferencia de datos a una PC. Esto provocó que el tamaño del dispositivo y el consumo de energía no fueron lo óptimo para un dispositivo tipo *wearable*.

Debido a lo anterior, se realizó un segundo diseño orientado a la clasificación en tiempo real. Para este diseño se trató de cumplir con las restricciones de tamaño y consumo de energía de un *wearable*. Con este diseño se pretende demostrar que el algoritmo de clasificación de actividades físicas o el de detección de caídas pueden funcionar en un dispositivo vestible cuya autonomía sea de varios días. A diferencia del anterior diseño, este se mando a fabricar a una empresa de manufactura de PCB, lo que permitió que la tarjeta tenga un menor tamaño al poder incluir componentes más pequeños y pistas más delgadas.

7.1. Diseño del dispositivo

En la figura 7.1 se puede ver un diagrama simple de la arquitectura de Hardware del dispositivo.

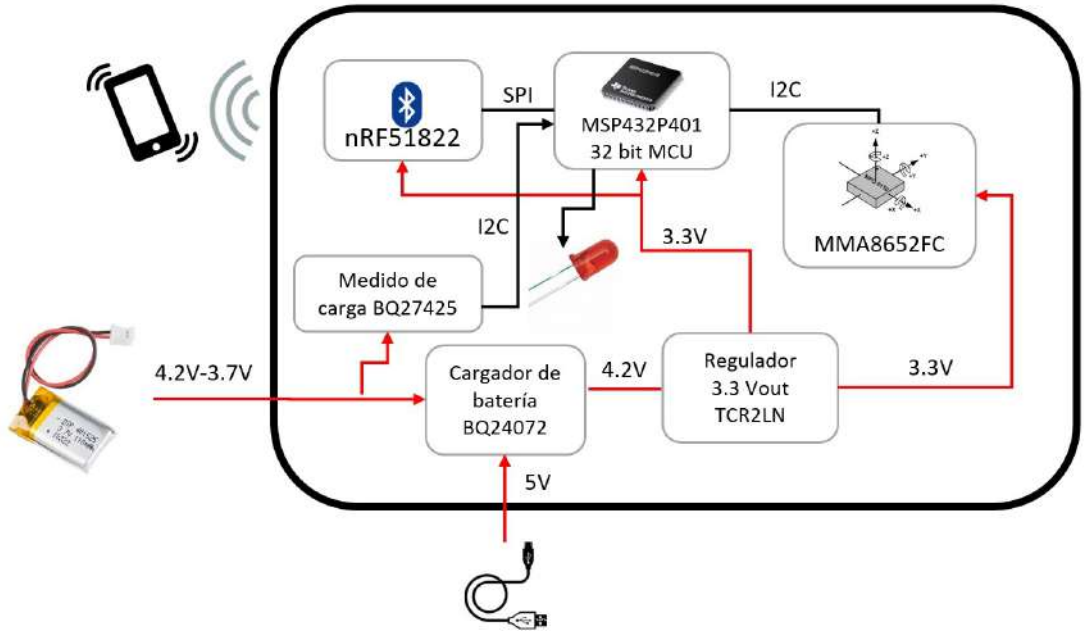

Figura 7.1: Diagrama de la arquitectura de Hardware del dispositivo

Como se puede apreciar, al contrastar el diagrama de la figura 7.1 con el de la figura 4.3, se realizaron varios cambios en el diseño del dispositivo. Entre los más significativos se encuentran los siguientes:

- Ya no se incluyen botones, diodos leds y la ranura para micro SD.
- Se añadió un módulo Bluetooth Low Energy 4.0.
- Se agregó un puerto micro USB y el circuito para carga de baterías tipo Li-Po o Li-Ion.
- Se actualizó el microcontrolador de la versión MSP430 a la MSP432.
- Los componentes se seleccionaron en sus versiones más pequeñas para optimizar el tamaño del PCB.
- Se añadió un acelerómetro de 3 ejes MMA8652FCR1.

A continuación se hará una descripción breve de los componentes principales de este diseño.

7.1.1. Alimentación del sistema

A diferencia del primer dispositivo en donde la batería debía cargarse de manera externa, para este segundo diseño se incluyo un circuito cargador de baterías.

El dispositivo se alimentará con una pequeña batería tipo Li-Po de 150 mAh. El circuito cargador permitirá la carga de esta a través de un puerto micro USB, utilizando una fuente de 5 Volts, como un cargador para teléfono móvil o un puerto USB de una computadora.

Este circuito consiste del integrado BQ24072 del fabricante *Texas Instruments*, el cual permite una carga de hasta 100 mA de corriente, por lo que la batería tarda aproximadamente 1 hora y media para cargarse completamente. Además, este integrado cuenta con una función para seleccionar la fuente de alimentación del sistema, ya sea el voltaje de la batería o el voltaje de carga si está conectado, esto sin interrumpir la operación del sistema al conectar o desconectar el cargador. La salida de este integrado es un voltaje de 4.2 Volts.

Para proveer el voltaje de 3.3 Volts de alimentación del sistema, se añadió un regulador TCR2LN de marca *TOSHIBA*. Este regulador es más pequeño, consume menos energía y es de un menor costo que el utilizado en la primera versión. Este mide solamente 0.8 x 0.8 x 0.3 mm y consume $2\mu A$.

7.1.2. Bluetooth Low Energy

Debido a que ya no se cuenta con botones, leds y la micro SD se agregó un módulo Bluetooth NRF51822 del fabricante *Nordic Semiconductors* para la transferencia de datos y la interacción con el usuario incluyendo la visualización de los resultados de la clasificación en tiempo real. El integrado NRF51822 es un microcontrolador con arquitectura ARM Cortex-M0 de 32 bits, en el cual puede integrarse el protocolo Bluetooth Low Energy 4.0. Utilizando este protocolo, el dipositivo puede establecer una conexión inalámbrica, de muy bajo consumo de energía, con otro dispositivo móvil.

La principal característica del NRF51822 es su bajo consumo de energía. Este puede llegar a consumir menos de $100\mu A$ mientras hay una conexión activa con otro dispositivo a través de Bluetooth y este se encuentra lo suficientemente cerca. Mientras está a la espera de una conexión, este debe anunciarse constantemente por lo que su consumo puede incrementarse hasta 800 μA lo que reduce la duración de la batería. Entre otras características, el NRF51822 cuenta con 128KB de memoria flash, 16kB de memoria RAM y su empaquetado mide 6x6 mm. Estas características lo hacen ideal para dispositivos “vestibles”.

7.1.3. Medidor de carga

Para que el usuario conozca el nivel de carga restante en la batería, se incluyó en el diseño el integrado BQ27425 del fabricante *Texas Instruments*. Con este circuito es posible conocer, además del porcentaje de carga restante, la carga máxima en mAh, la cantidad de corriente con la que se está cargando o descargando en mA, el voltaje y la salud de la batería.

7.1.4. Microcontrolador MSP432

El microcontrolador MSP430f5529 fue sustituido para esta segundo diseño por una versión más actual de la misma línea de microcontroladores, el MSP432P401. El MSP432 tiene una arquitectura ARM Cortex-M4F de 32-bits, a diferencia del MSP430 de 16-bits usado anteriormente.

La velocidad del procesador del MSP432 puede alcanzar una frecuencia de hasta 48Mhz, cuenta con más memoria ram y flah (64KB y 256KB respectivamente) que el MSP430, tiene un consumo de enrgía en modo activo de 80 μA/Mhz y además, cuenta con un módulo de hardware para operaciones de punto flotante. Esta última característica le permite al microcontrolador realizar operaciones con punto flotante en una cantidad de tiempo mucho menor que el MSP430 y por lo tanto consume menos energía en aplicaciones en donde se requiera ejecutar una gran cantidad de operaciones de este tipo.

7.1.5. Acelerómetro

Después de analizar los resultados obtenidos de las pruebas, se optó por cambiar el sensor inercial MPU-9150 por un sensor que solo cuenta con acelerómetros. La principal razón fue que se consideró que la mejora en el rendimiento obtenido en la clasificación de actividades al utilizar datos del giroscopio en el vector de características no fue suficiente para compensar el consumo de energía del sensor al tener activado el giroscopio y, el costo del mismo.

El sensor utilizado para esta versión fue el MMA8652FC del fabricante *Freescale*, el cual cuenta con acelerómetros de 12 bits de resolución en los 3 ejes y alcanza una frecuencia de muestreo de hasta 800 Hz.

En cuanto al consumo de energía, el MPU-9150 de la primera versión consumía 3.9 mA si se utilizaba el giroscopio, por lo que la batería de 150μA utilizada para esta versión tendría una duración de menos de 2 días. El consumo de energía del MMA8652FC es de 15uA con una frecuencia de muestreo de 50 Hz . En cuanto al costo, el MPU-9150 tiene un precio de 8 dólares mientras que el precio

del MMA8650FC es de 0.83 dólares. En esta comparación se aprecia por qué se decidió no utilizar el giroscopio para la clasificación de actividades a pesar de la disminución en el rendimiento del algoritmo.

7.2. Diseño esquemático del dispositivo

Utilizando el diseño esquemático realizado en el software Cadsoft Eagle se describirán brevemente las conexiones de los componentes y las señales entre estos. El diseño lo podemos encontrar de la figura 7.2 a la figura 7.6.

En la figura 7.2 se encuentra el conector micro USB (**U3**) en donde se conecta una fuente externa de 5V. Cuenta con un diodo (D1) para evitar un voltaje inverso que pueda dañar el dispositivo. **J1** es el conector de la batería y **U2** es el integrado que controla la carga de esta. Este recibe el voltaje de la batería y el voltaje de la fuente externa, si está presente, y su salida (**V_CHRGR_OUT**) es de 4.2 Volts. También cuenta con una señal (**CHRGR_CE**) para habilitar o deshabilitar la carga de la batería.

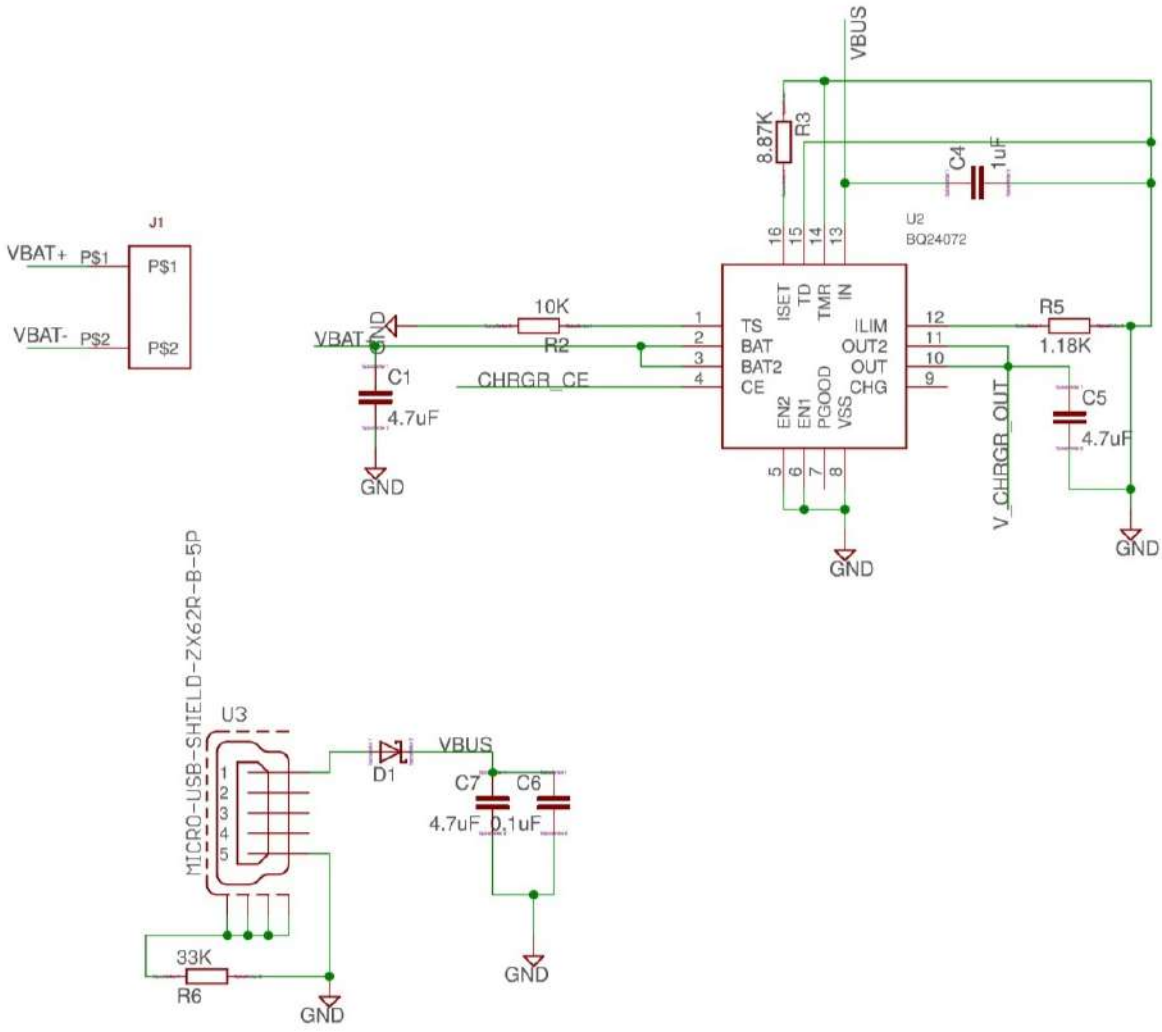

Figura 7.2: Diseño esquemático de la segunda versión del dispositivo (1/5).

En la figura 7.3 se encuentra el regulador de voltaje (U1) que alimenta al sistema. Su entrada es el voltaje de salida del cargador de baterías, la señal **V_CHRGR_OUT** de 4.2 Volts y su salida es un voltaje de 3.3 Volts. **U4** es el medidor de carga de la batería, el cual se comunica con el microcontrolador a través del protocolo **I2C**.

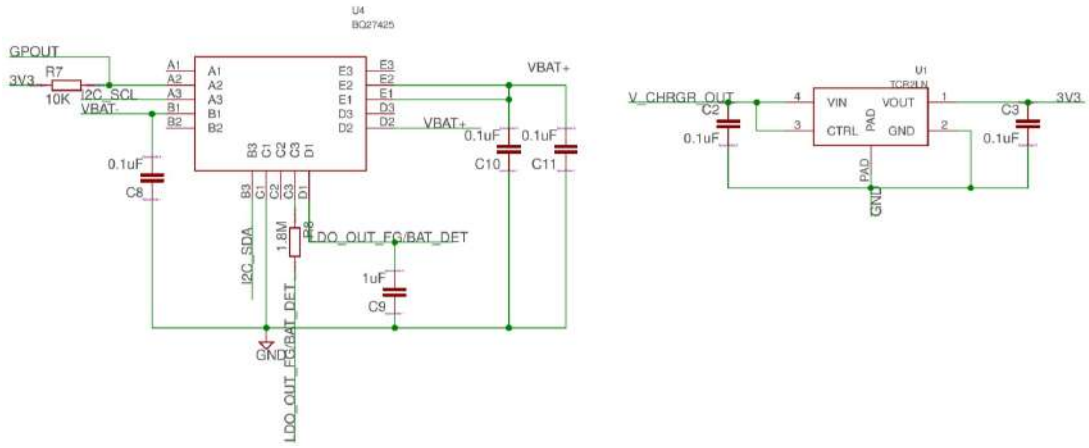

Figura 7.3: Diseño esquemático de la segunda versión del dispositivo (2/5).

En la figura 7.4 se encuentra el circuito del microcontrolador MSP432 (**U6**). Para funcionar correctamente solamente necesita de capacitores de desacoplo, un inductor y un resistor. **Y1** y **Y2** son cristales de reloj necesarios para mejorar la estabilidad y precisión de las fuentes de reloj del procesador.

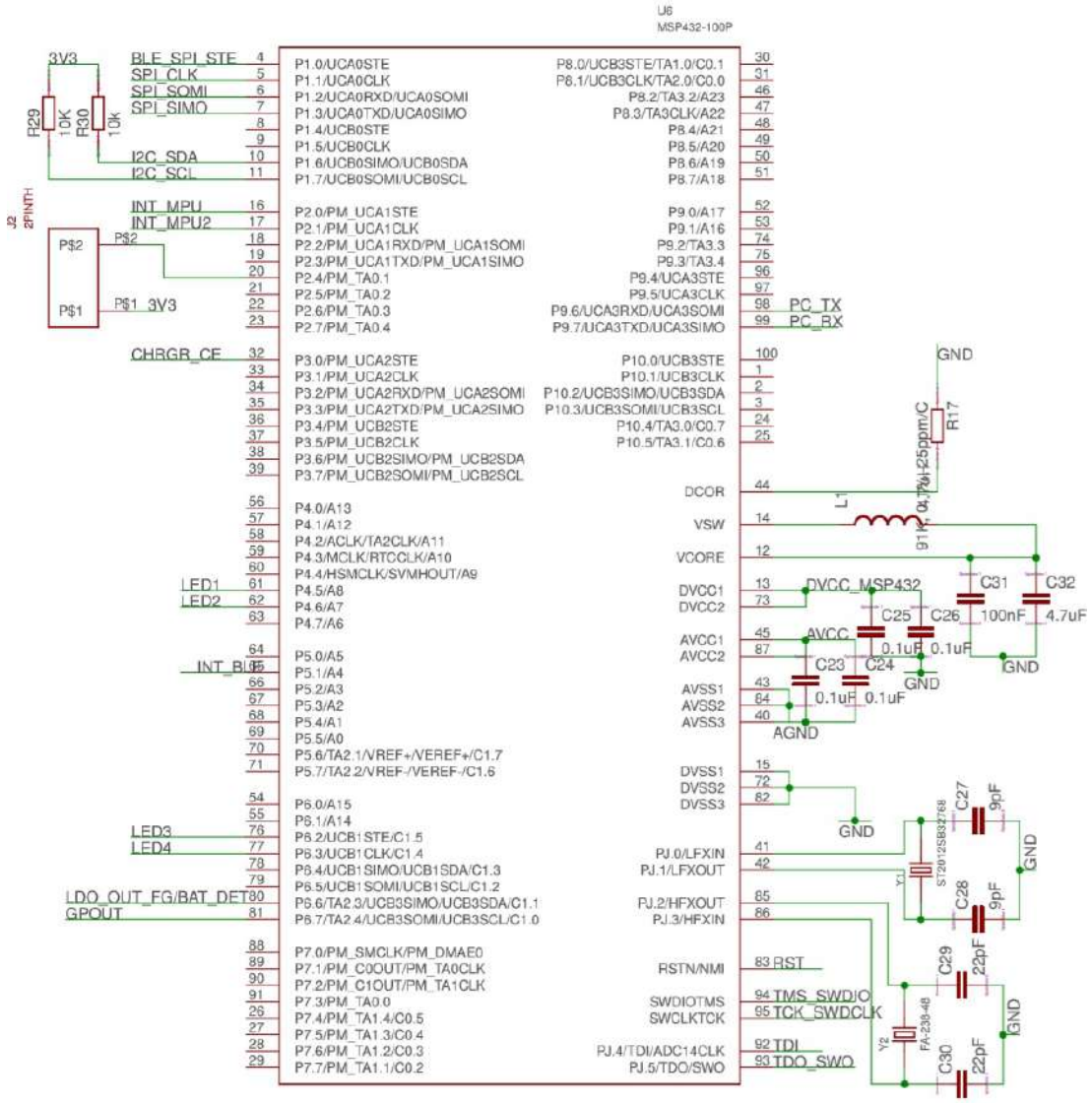

Figura 7.4: Diseño esquemático de la segunda versión del dispositivo (3/5).

En la figura 7.5 **U5** es el sensor acelerómetro, este se comunica con el microcontrolador utilizando el protocolo **I2C** para el intercambio de datos y dos señales de interrupción (**INT_MPU** e **INT_MPU2**) para indicar diferentes eventos en el funcionamiento del sensor. Como componentes externos, solamente necesita de capacitores de desacoplo. En la misma figura, **J3** es un conector con las señales necesarias para depurar o cargar código en el microcontrolador y en el Bluetooth. **D1** y **D2** son diodos LED que se utilizarán para indicar al usuario si la carga de la batería es baja y el estado de la conexión con otro dispositivo Bluetooth.

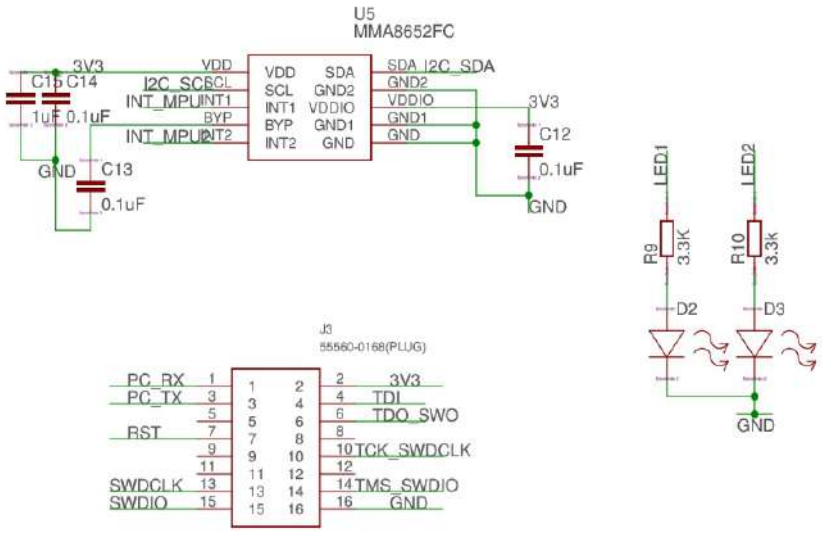

Figura 7.5: Diseño esquemático de la segunda versión del dispositivo (4/5).

Por último, en la figura 7.6 se puede ver el circuito del módulo Bluetooth NRF51822 (**U7**). Este circuito corresponde al recomendado en la hoja de datos para la utilización de una antena de PCB (**A1**). Esto debido a que el tamaño del dispositivo hace difícil poder añadir una antena externa. Se utiliza el protocolo serial SPI y una señal de interrupción (**INT_BLE**) para el intercambio de datos entre el microcontrolador y el módulo Bluetooth.

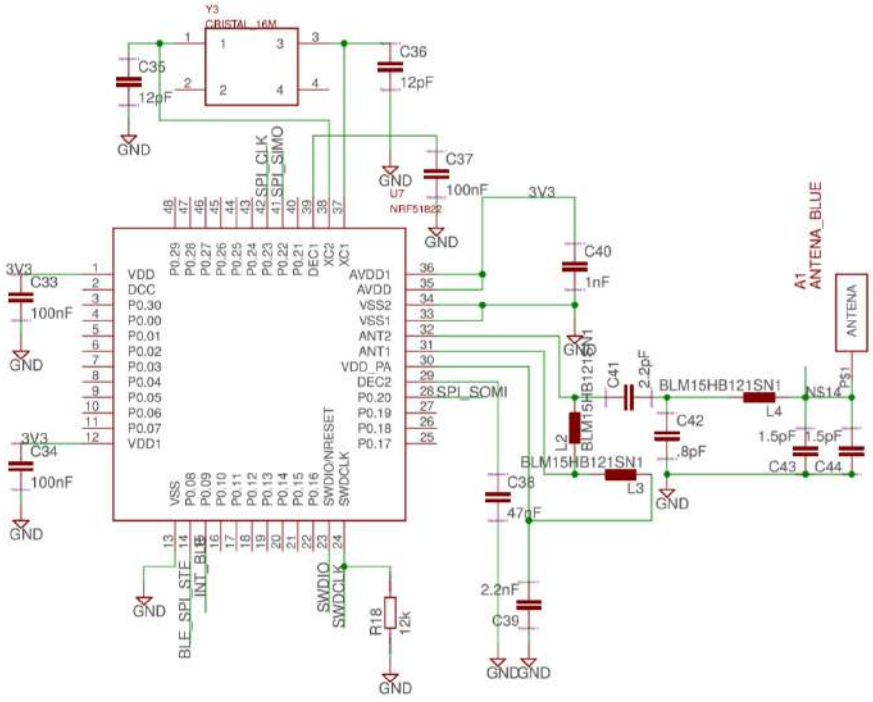

Figura 7.6: Diseño esquemático de la segunda versión del dispositivo (5/5).

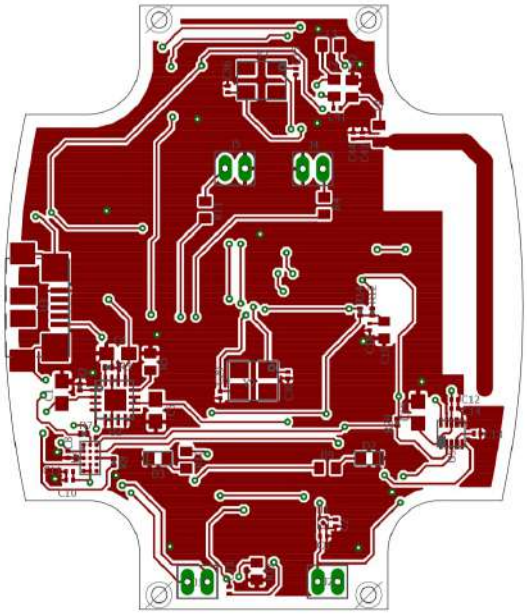

Figura 7.7: Capa superior del layout del dispositivo.

7.3. Fabricación del dispositivo

El diseño *Layout* de la tarjeta de circuito impreso se realizó en el software Cadsoft Eagle. En las figuras 7.7 y 7.8 podemos ver la capa superior e inferior respectivamente del diseño. Debido al tamaño de los componentes y del grosor de las pistas utilizado, la tarjeta y el montaje de los componentes se mando a hacer en una empresa especializada en fabricación de PCB.

La tarjeta fue diseñada de tal forma que encajara en un encapsulado que se adquirió, el cual podemos ver en la figura 7.9. En las figuras 7.10, 7.11 y 7.12 podemos ver fotos de la electrónica y del dispositivo final. Como se había mencionado, el tamaño y consumo de energía de esta versión del dispositivo si cumple con las restricciones de un dispositivo *wearable*.

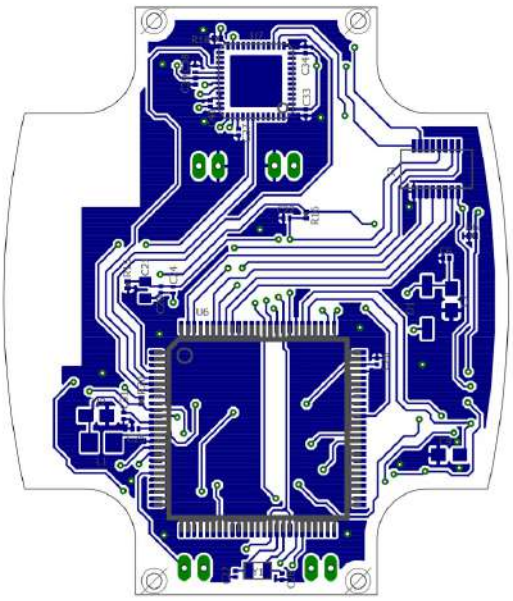

Figura 7.8: Capa inferior del layout del dispositivo.

Figura 7.9: Encapsulado en donde se coloca la electrónica.

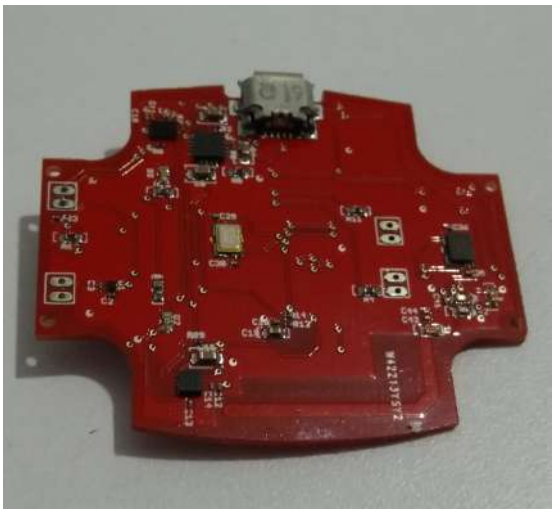

Figura 7.10: Foto de la tarjeta electrónica.

Figura 7.11: Foto de las partes dispositivo final.

Figura 7.12: Foto del dispositivo final.

Capítulo 8

Conclusiones

A pesar de que en la literatura podemos encontrar una gran cantidad de trabajos acerca de la detección automática de actividades, muy pocos de ellos analizan su implementación en tiempo real en dispositivos embebidos de bajos recursos, como lo son los dispositivos *wearables* o vestibles. En este trabajo de tesis se desarrollo un sistema de clasificación de actividades físicas que puede ejecutarse en un microcontrolador de bajos recursos. Se mostró que utilizando una Red Neuronal y características simples se puede obtener un clasificador de actividades con una precisión aceptable, y que al ser, el algoritmo clasificador resultante, de baja complejidad computacional, su consumo de recursos y de energía es suficientemente bajo para portarse en un dispositivo *wearable*.

También, se mostró que el dispositivo y los programas que se hicieron para el desarrollo del clasificador de actividades físicas pueden fácilmente ser adaptados para la clasificación de otros tipos de actividades en los que un sensor inercial pueda proporcionar información relevante, como se realizó con la detección de caídas.

El sistema de clasificación de actividades desarrollado solo funcionará cuando el usuario esté realizando alguna de las 6 actividades que es capaz de detectar. Como trabajo futuro estaría el desarrollo de una solución para el problema de la clase nula, que debería ser la salida del clasificador cuando no detecte ninguna de las actividades con las que fue entrenado. Otra función que se podría añadir al sistema sería el entrenamiento en tiempo real, es decir, que el algoritmo vaya mejorando su rendimiento con el uso del dispositivo con base a la retroalimentación que pudiera obtener del usuario. De esta misma manera, el usuario podría agregar la detección de actividades nuevas al clasificador.

Bibliografía

[1] Nagendra B. Bharatula, Mathias Stäger, Paul Lukowicz, and Gerhard Tröster. Power and size optimized multi-sensor context recognition platform. *Proceedings - International Symposium on Wearable Computers, ISWC*, 2005:194–195, 2005.

[2] Bingbing Ni, GangWang and PierreMoulin. RGBD-HuDaAct: A Color-Depth Video Database for Human Daily Activity Recognition. pages 193–208, 2017.

[3] Tanzeem Choudhury, Gaetano Borriello, Sunny Consolvo, Dirk Haehnel, Beverly Harrison, Bruce Hemingway, Jeffrey Hightower, Predrag Klasnja, Karl Koscher, Anthony LaMarca, James A. Landay, Louis LeGrand, Jonathan Lester, Ali Rahimi, Adam Rea, and Danny Wyatt. The mobile sensing platform: An embedded activity recognition system. *IEEE Pervasive Computing*, 7(2):32–41, 2008.

[4] Donghai Guan, Weiwei Yuan, Young-koo Lee, Andrey Gavrilov, and Sungyoung Lee. Activity Recognition Based on Semi-supervised Learning. (1):1–7, 2007.

[5] Abdelsalam Helal, Diane J. Cook, and Mark Schmalz. Smart Home-Based Health Platform for Behavioral Monitoring and Alteration of Diabetes Patients. *Journal of Diabetes Science and Technology*, 3(1):141–148, 2009.

[6] Luciana C Jatobá, Ulrich Großmann, Chistophe Kunze, Jörg Ottenbacher, Wilhelm Stork, and A Measurement System. Pattern Recognition Methods for Classification of Physical Activity. pages 5250–5253, 2008.

[7] Holger Junker, Oliver Amft, Paul Lukowicz, and Gerhard Tröster. Gesture spotting with body-worn inertial sensors to detect user activities. 41(2008):2010–2024, 2010.

[8] Zafar A. Khan and Won Sohn. Abnormal human activity recognition system based on R-transform and kernel discriminant technique for elderly home care. *IEEE Transactions on Consumer Electronics*, 57(4):1843–1850, 2011.

[9] Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity Recognition using Cell Phone Accelerometers. 12(2):74–82, 2010.

[10] Yongjin Kwon, Kyuchang Kang, and Changseok Bae. Expert Systems with Applications Unsupervised learning for human activity recognition using smartphone sensors. *EXPERT SYSTEMS WITH APPLICATIONS*, (May), 2014.

[11] Oscar D. Lara and Miguel A. Labrador. A Survey on Human Activity Recognition using Wearable Sensors. *IEEE Communications Surveys & Tutorials*, 15(3):1192–1209, 2013.

[12] Óscar D. Lara, Alfredo J. Prez, Miguel A. Labrador, and Jos D. Posada. Centinela: A human activity recognition system based on acceleration and vital sign data. *Pervasive and Mobile Computing*, 8(5):717–729, 2012.

[13] Lin Liao, Dieter Fox, and Henry Kautz. Location-based Activity Recognition.

[14] U. Maurer, A. Smailagic, D.P. Siewiorek, and M. Deisher. Activity recognition and monitoring using multiple sensors on different body positions. *International Workshop on Wearable and Implantable Body Sensor Networks (BSN'06)*, pages 4–7, 2006.

[15] Futoshi Naya, Ren Ohmura, Fusako Takayanagi, Haruo Noma, and Kiyoshi Kogure. Workers' Routine Activity Recognition using Body Movements and Location Information. *2006 10th IEEE International Symposium on Wearable Computers*, pages 105–108, 2006.

[16] Jie Yin, Qiang Yang, and Jeffrey Junfeng Pan. Sensor-based abnormal human-activity detection. *IEEE Transactions on Knowledge and Data Engineering*, 20(8):1082–1090, 2008.