UNIVERSIDAD DE VALLADOLID

ESCUELA TÉCNICA SUPERIOR

INGENIEROS DE TELECOMUNICACIÓN

TRABAJO DE FIN DE GRADO

GRADO EN TECNOLOGÍAS ESPECÍFICAS DE TELECOMUNICACIÓN:
MENCIÓN EN SISTEMAS DE TELECOMUNICACIÓN

Diseño e Implementación de Redes Neuronales de Aprendizaje Profundo para Clasificación y Análisis de Movimientos Corporales Capturados Mediante Dispositivos Vestibles

Autor:

D. David Arévalo González

Tutores:

Dr. D. Mario Martínez Zarzuela
Dra. Dª. Cristina Simón Martínez

Valladolid, Septiembre 2021

TÍTULO:	**Diseño e Implementación de Redes Neuronales de Aprendizaje Profundo para Clasificación y Análisis de Movimientos Corporales Capturados Mediante Dispositivos Vestibles**
AUTOR:	**D. David Arévalo González**
TUTORES:	**Dr. D. Mario Martínez Zarzuela** **Dra. Dª. Cristina Simón Martínez**
DEPARTAMENTO:	**Departamento de Teoría de la Señal y Comunicaciones e Ingeniería Telemática**

Tribunal

PRESIDENTE:	**Dr. D. Francisco Javier Díaz Pernas**
VOCAL:	**Dr. D. David González Ortega**
SECRETARIO:	**Dr. D. Mario Martínez Zarzuela**
SUPLENTE 1:	**Dra. Dª. Míriam Antón Rodríguez**
SUPLENTE 2:	**Dr. D. Carlos Gómez Peña**
FECHA:	**Septiembre 2021**
CALIFICACIÓN:	

Agradecimientos

A mi familia y amigos, por ayudarme a ser la persona que soy hoy y ser un apoyo incondicional a lo largo de toda mi vida, pero sobre todo en estos últimos años.

A mis tutores y, en especial a Mario, por la gran ayuda, motivación y apoyo que me ha prestado en la realización de este trabajo. A toda la gente que me ha ayudado durante el desarrollo de este trabajo directa o indirectamente.

Resumen

Este Trabajo de Fin de Grado se centra en el problema del *Reconocimiento de Actividades Humanas* o *HAR* empleando *Redes Neuronales de Aprendizaje Profundo*, que se encuentran dentro de la *Inteligencia Artificial* y a su vez, del *Aprendizaje Automático*.

En nuestro caso concreto, los datos se han obtenido a través de *Sensores Inerciales* o *IMUs*, los cuales registran cuaterniones, indicando la orientación de la parte del cuerpo donde están situados. Con estas grabaciones, se ha llevado a cabo el procesado de los datos y la formación una base de datos con estas grabaciones procesadas. Partiendo de esta base de datos, y empleando redes convolucionales, hemos conseguido llevar a cabo el reconocimiento de actividades humanas. El resultado es la identificación de 13 actividades tanto de tren superior como de tren inferior con gran precisión.

Palabras Clave

Reconocimiento de Actividades Humanas, Sensores Inerciales, Base de Datos, Aprendizaje Profundo, Redes Neuronales Convolucionales.

Abstract

This project is focused on the *Human Activity Recognition* or *HAR* problem using *Deep Neural Networks*, enclosed in *Deep Learning* and so they are in *Machine Learning*.

In our case, data was obtained with *Inertial Sensor Units* or *IMUs*, which register quaternions, indicating the orientation of the body part they are placed on. These recordings were processed and used to build a database. Using this database and Convolutional Neural Networks, We were successful to achieve the targeted activity recognition. The result was the precise recognition of 13 different activities from both the upper and lower part of the body.

Keywords

Human Activity Recognition, Inertial Sensor Units, Database, Deep Learning, Convolutional Neural Networks.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Este capítulo pretende servir como una descripción del marco de trabajo y desarrollo de este documento, así como sus objetivos, métodos y resultados obtenidos a lo largo de la realización del mismo.

El presente Trabajo de Fin de Grado se ha llevado a cabo bajo la tutorización del Dr. D. Mario Martínez Zarzuela, perteneciente al *Grupo de Telemática e Imagen* (GTI) dentro de la *Escuela Técnica Superior de Ingenieros de Telecomunicación* (ETSIT) de la *Universidad de Valladolid* (UVa). Las líneas de investigación principales que sigue Mario se centran en el uso de sensórica, tanto infrarroja como sensores inerciales (IMU, *Inertial Measurement Unit*) empleando sistemas de Aprendizaje Automático (ML) o de Aprendizaje Profundo (DL). Los objetivos finales de sus investigaciones pueden ser evaluar la rehabilitación de pacientes con enfermedades que implican dificultad de movimiento, mejorar la calidad de estas rehabilitaciones en entornos no clínicos y más cómodos para los pacientes, valoración de la ergonomía en puestos de trabajo, etc. También se ha contado con la co-tutorización de la Dra. D.ª Cristina Simón Martínez, fisioterapeuta especializada en rehabilitación basada en entrenamientos cognitivos y nuevas técnicas, neurorehabilitación, biomecánica motora y medicina personalizada en el *Haute Ecole Spécialisée de Suisse Occidentale HES SO Valais*, de la Universidad de Ciencias Aplicadas y Artes de Suiza Oeste. La Dra. Cristina ha aportado su ayuda y sus conocimientos en lo relativo a los aspectos de biomecánica, y conocimientos de rehabilitación que empleamos en el trabajo.

Este proyecto se comenzó en febrero de 2021 y se ha llevado a cabo mediante contactos frecuentes con los tutores a través de correo electrónico, reuniones telemáticas y para el desarrollo práctico del apartado 3.2.4 se emplearon reuniones presenciales a lo largo de una semana para hacer las capturas pertinentes con el fin de la creación de la base de datos propia empleada en este trabajo junto a varios estudiantes, los cuales estaban desarrollando sus respectivos trabajos de fin de estudios.

1.1. Contexto y Motivación

La última tendencia, justificada con la fácil accesibilidad a nuevas tecnologías por el abaratamiento del hardware, es la digitalización. La digitalización es el empleo de herramientas de medida y control para mecanizar o automatizar acciones, procesos o cualquier actividad o simplemente objetivizarlos que, hasta ahora se habían llevado a cabo de manera manual o con intervención directa humana. Puede tener diferentes objetivos, como mejorar la comodidad, optimización de tiempos o facilitar tareas con el fin de ahorrar esfuerzos, tiempo y dinero. En definitiva, es una de las aplicaciones directas de los axiomas de la tecnología y la ingeniería.

En la última década, los sistemas de aprendizaje automático y, en concreto, de aprendizaje profundo, han sufrido un desarrollo considerable. Todo ello ha sido provocado por tres agentes:

- **Hardware de procesamiento más potente:** como unidades de procesamiento gráfico (Graphic Processing Units, GPUs) o unidades de procesamiento tensorial (Tensorial Processing Units, TPUs) y desarrollo de lenguajes de programación, como Python o R, así como librerías específicas para el procesado de grandes cantidades de datos.
- **Desarrollo del Big Data:** creación de bancos, repositorios y bases de datos específicas para problemas de deep learning. Aumento de la capacidad de recogida de datos y generación de información útil. Los avances en este campo han sido beneficiados por el desarrollo de los sistemas del *Internet de las Cosas o IoT*.
- **Avances en el desarrollo de técnicas de Deep Learning:** En la aceptación comercial de nuevas tecnologías se siguen diferentes fases, recogidas en lo que se conoce como ciclo de sobreexplotación, que es un término creado por la empresa *Gartner* (Figura 1). Este ciclo es una representación gráfica de la madurez, adopción en el mercado y aplicación comercial efectiva [1]. El estado actual del Deep Learning se sitúa en el comienzo de la estabilización de la curva, esta zona se conoce como rampa de consolidación, y es el estado de la tecnología en cuestión previo a la zona llamada meseta de productividad.

El desarrollo conjunto de estos 3 ítems ha desencadenado que vuelva el interés en este campo de investigación para el diseño de sistemas de aprendizaje profundo empleados en tareas en las que la programación clásica no alcanza o se queda corta y mejorar de manera considerable todas las métricas que se han conseguido empleando técnicas de aprendizaje automático clásico.

También en la última década, el problema del *reconocimiento de actividades* (*Human Activity Recognition*) o HAR ha cobrado especial atención. Esto se debe a la numerosa cantidad de aplicaciones que han comenzado a surgir por el proceso de digitalización en numerosos y diversos campos. Otro motivo para ello es la mejora en la precisión y en la eficiencia de los sistemas empleados en la materia para reconocer acciones ejecutadas por los usuarios o pacientes.

La clave de trabajar el HAR es que los movimientos del cuerpo humano, en la realización de cualquier actividad se generan patrones. Estos patrones son fácilmente medibles con sistemas como sensores o cámaras y, por lo tanto, son clasificables por algoritmos de *Machine Learning*.

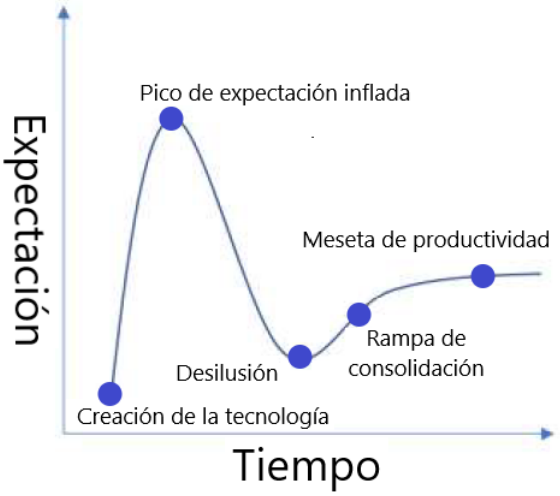

Figura 1: Ciclo de sobreexplotación

En la actualidad, el reconocimiento de actividades humanas tiene como principales objetivos las aplicaciones en entornos médicos, como evaluaciones o rehabilitaciones de pacientes, preventivas en sujetos sanos; en entornos deportivos para una mejora técnica y su evaluación; o en factorías para el control de la ergonomía. El entorno médico es uno de los campos de aplicación más relevantes, pero está aumentando la demanda de este tipo de sistemas en entornos no clínicos [2]. Estamos en pleno auge de los *smartphones*, los cuales cuentan con multitud de sensores. Varios de estos sensores ya se emplean actualmente para llevar a cabo HAR, pero a un nivel menos preciso y exhaustivo como requieren las aplicaciones médicas actuales. También, en un futuro próximo, gracias a la mejora de estos sensores, podremos disponer de métricas sobre nuestra actividad corporal sin necesidad de preocuparnos por el proceso de medición.

Comúnmente, los sistemas HAR constan de las siguientes etapas:

1. **Medición:** Es el proceso de obtención de datos sobre las actividades desarrolladas por los sujetos que posteriormente se utilizarán para la identificación de las propias actividades. Dependiendo del sistema de captación de esta información, se pueden obtener , vídeo, profundidades (vídeo con sensores RGBD), imágenes, aceleraciones, posiciones, etc.
2. **Segmentación de los datos:** Este paso se centra en llevar a cabo una primera limpieza de los datos, separando las grabaciones de manera que se puedan identificar las actividades que se desean clasificar individualmente. También es necesario que se realice una eliminación de artefactos y/o ruido para dejar la información útil que vayamos a poder usar.
3. **Extracción de características:** Esta fase del proceso puede ser opcional en función del posterior método de clasificación que empleemos. Las características son mediciones o cálculos aplicados a las grabaciones previamente tratadas. Estas características pueden ser de diferentes tipos atendiendo a la manera en la que los calculemos:
    - Características temporales: Son las calculadas a partir de una secuencia temporal.
    - Características frecuenciales: Son las calculadas a partir de una estimación del espectro de la señal. Estas estimaciones pueden calcularse mediante la FFT (Fast

Fourier Transform), la STFT (Short-Time Fourier Transform), la CWT (Continuous Wavelet Transform)... en función de las características de la señal en el dominio temporal.

Algunas de estas características pueden ser el valor medio, la varianza de los datos, correlación cruzada con otras señales, la frecuencia mediana, la frecuencia media, la frecuencia a la que se da el máximo de la energía de la señal, el ancho de banda que ocupa tanto porcentaje del espectro, entre muchos otros.

4. **Clasificación:** La clasificación es la parte más crítica del proceso. Actualmente es uno de los campos de investigación en auge, ya que se emplean algoritmos y técnicas de *Machine Learning* y *Deep Learning*, las cuales están recibiendo mucha atención por los resultados tan prometedores que se están consiguiendo en las publicaciones más recientes [2]. Esta fase consiste en la identificación automática de los datos capturados tras un proceso de entrenamiento de las redes o sistemas que empleemos. Para este entrenamiento se requiere una gran potencia computacional, ya que para obtener buenos resultados es necesario emplear grandes cantidades de información y muchas iteraciones en los entrenamientos [3].
5. **Post-procesado:** Esta fase del proceso depende de cómo se desee mostrar la información sobre el resultado de la clasificación que se ha obtenido. Puede que se requiera representar la información a través de un display o que se trabaje con un software en tiempo real que muestra la información de manera más gráfica, etc.

En lo relativo a la medición dentro del proceso de HAR, hay dos grandes tipos de maneras de abordarla:

- Sistemas basados en imagen, empleando sensores fotoeléctronicos.
- Sistemas basados en sensórica vestible.

Los sistemas que emplean imágenes, recogen datos de sensores como cámaras, videocámaras, sensores RGBD, láseres, etc. Estos sistemas tienen la gran ventaja de que el usuario no requiere llevar puesto ningún complemento, sensor o accesorio para la captación de los datos, pero vulneran la privacidad de los usuarios y estos sistemas se ven muy afectados por las condiciones ambientales de la captación, como la luz, ruido de radiofrecuencia, posición inadecuada del dispositivo de captación o del sujeto del que adquirir los datos, entre otros.

En la otra cara de la moneda se encuentran los sistemas que emplean sensores vestibles. Estos sensores son capaces de captar aceleraciones triaxiales, velocidades angulares, presiones o incluso de recogida de señales biomédicas como el electromiograma (EMG), que mide la respuesta eléctrica de los músculos. La ventaja que tienen es que la calidad de los registros (por lo general) no depende de las condiciones ambientales mencionadas en el apartado anterior y las mediciones son muy precisas. También, la transmisión de los datos puede hacerse tanto de manera alámbrica, lo cual limita la movilidad y comodidad de la captura de los datos; o inalámbrica, la cual supone una mayor comodidad, aunque se pueden sufrir las consecuencias de las interferencias de radiofrecuencia con otros sistemas o protocolos de comunicación que

compartan la misma banda de frecuencia. La desventaja de estos sistemas es que pueden generar una mayor incomodidad al usuario, ya que es necesario que éste porte los sensores para llevar a cabo la captación,

Como en muchas situaciones, estos sistemas de captación no son excluyentes, y permiten la utilización simultánea de varios, así podemos emplear los sistemas de visión como ayuda para los sistemas vestibles y así conseguir mejorar el desempeño de la clasificación.

En nuestro caso, emplearemos un tipo de sensores vestible llamados sensores inerciales (*Inertial Measurement Units, IMUs*), ya que, según las ideas expuestas, son las que nos aportan más ventajas en el escenario que deseamos trabajar. Este escenario es el análisis de HAR en sujetos sanos por las proyecciones de futuro de este campo investigación hacia un mercado no médico o clínico. Por ello, emplearemos sensores o unidades de medida inercial (IMUs) que han sido desarrollados por el doctorando del grupo de investigación GTI D. Javier González Alonso [4, 5] bajo el nombre de *TwynSens*.

Sensores TwynSens

Estos IMUs emplean sensores *BNO080 de Bosch*. Las mediciones que obtenemos de él en su salida son orientaciones absolutas, velocidades angulares, aceleraciones angulares y aceleraciones lineales a una tasa de hasta 100Hz, medidas sobre el campo magnético a una tasa de 20Hz y la gravedad y la temperatura a una tasa de 1Hz. De todas estas salidas, sólo capturaremos las orientaciones absolutas en forma de cuaterniones. Los sensores emplean un protocolo de comunicaciones inalámbrico, mediante Bluetooth, por lo que usa la banda de los 2.4GHz, lo cual puede suponer una ventaja dependiendo de la ocupación del espectro radioeléctrico por otras tecnologías.

Inicialmente, estos sensores estaban pensados para la valoración ergonómica y la evaluación clínica, pero los fines con los que los utilizaremos no son un motivo para descartar su uso. Las mediciones y pruebas llevadas a cabo en [5] demuestran la validez y fiabilidad de las mediciones llevadas a cabo con estos sensores.

1.2. Hipótesis y Objetivos

La hipótesis de este proyecto es, que partiendo de grabaciones capturadas con sensores inerciales desarrollados en [5], podemos llevar a cabo el reconocimiento de las actividades desarrolladas por unos sujetos empleando redes de aprendizaje automático y de aprendizaje profundo diseñadas por nosotros mismos.

Para alcanzar los objetivos expuestos, este trabajo ha seguido las siguientes fases:

- Crear una base de datos de actividades de la vida cotidiana llevadas a cabo por sujetos

sanos, no padecientes de ninguna patología en formato de cuaterniones provenientes de IMUs.

- Procesar de los datos capturados para obtener mediciones angulares útiles.
- Llevar a cabo una comparativa entre nuestro método de adquisición de datos con un sistema de captación basado en vídeo y comprobar la veracidad y fiabilidad de las métricas angulares empleadas.
- Llevar a cabo diferentes pruebas iniciales diseñando diferentes redes neuronales para clasificación de actividades de tronco superior y tronco inferior.

Por aclarar lo expuesto, el objetivo final de este proyecto es crear un sistema como el que se muestra de manera gráfica en la figura 2.

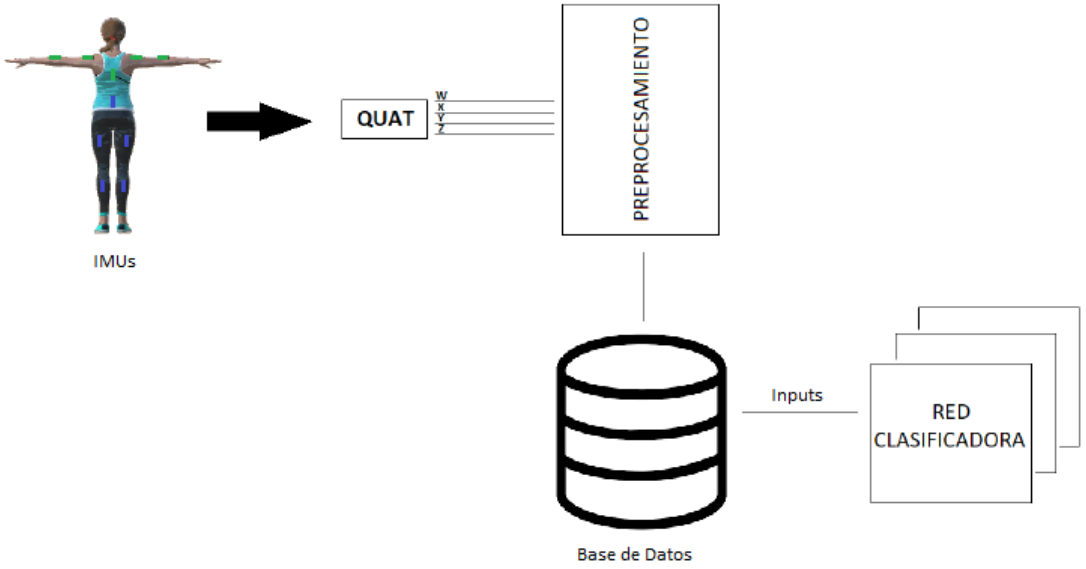

Figura 2: Diagrama del sistema propuesto

1.3. Fases y Métodos

Para el desarrollo del trabajo, se han seguido el siguiente orden de actividades:

1. **Adquisición de conocimientos sobre Machine Learning:** Para este trabajo, se partía del conocimiento nulo sobre inteligencia artificial. Por ello, lo primero que se hizo fue completar el curso de Machine Learning impartido por Andrew Ng a través de la plataforma *Coursera* [3] para una introducción bastante extendida sobre conceptos básicos y técnicas de Machine Learning clásico.
2. **Adquisición de conocimientos sobre Deep Learning:** Para expandir los conocimientos en la rama más específica que incumbía al trabajo de una manera práctica, se realizó el

curso *Fundamentals of Deep Learning for Computer Vision* de *NVIDIA Deep Learning Institute* [6]. Finalmente, se empleó el libro [7] para acercarse a la situación en la que se iba a desarrollar y entrar en materia sobre las posibles soluciones que se pueden aplicar con Deep Learning.

3. **Adquisición de conocimientos sobre Python:** Para este trabajo se partió del conocimiento nulo sobre Python, la única experiencia con la programación anterior al trabajo ha sido en las asignaturas cursadas a lo largo de la carrera, empleando C, HTML, CSS, JavaScript, Java y MatLab. Para poder entrar en contacto con el lenguaje de programación Python se leyó el libro [8].
4. **Recolección de información sobre IMUs:** Lectura de documentos técnicos sobre Xsens DOT [9], y asistencia al *Webinar: Xsens DOT Developer Conference: Automated Manual Tasks Risk Assessment, a benefit or a distraction?* [10] donde empresas internacionales exponen diferentes necesidades, productos y servicios que desarrollan con los sensores inerciales Xsens DOT. De esta manera podemos ver las aplicaciones más novedosas que se están dando a estos dispositivos. La recolección de datos para la posteriormente mencionada y explicada base de datos se emplearon sensores no comerciales similares a los Xsens DOT desarrollados por el doctorando D. Javier González Alonso [4, 5]
5. **Conformación de la base de datos propia:** Esta fase se desarrollará con mayor profundidad en el apartado 3.1.
6. **Preprocesamiento de los datos:** Esta fase se explica en el apartado 3.1.5 y consiste en el tratamiento que ha de dársele a los datos crudos de las grabaciones para poder extraer información útil de ellos.
7. **Investigación e implementación de algoritmos para obtención de ángulos a partir de los cuaterniones registrados:** Se emplearon ángulos de Euler y desplazamientos angulares. La explicación sobre esta fase se explica en los apartados 3.2.3, 3.2.4 y 3.2.5.
8. **Realización de una revisión del Estado del Arte en HAR, DL e IMUs:** Para saber cuál es la situación actual en la investigación de HAR, es necesario hacer una revisión de la literatura publicada en los últimos años que traten temas o hagan experimentos relacionados con los que se pretenden desarrollar en este TFG.
9. **Realización de la comparativa con sistema de captura de datos mediante vídeo:** Para el desarrollo de este apartado, tuvimos que llevar a cabo una reunión con D. Diego Pérez de la Fuente, autor de [11], un trabajo de investigación sobre comparativas entre diferentes sistemas de captura de datos basados en imagen y vídeo con el fin de conseguir el reconocimiento de actividades. El objetivo de la reunión fue poner ideas en común sobre la estructura, qué contenidos incluir y los puntos importantes que mencionar en la comparación.
10. **Aprendizaje sobre el *framework* de redes neuronales:** Se llevaron a cabo diferentes reuniones con D. Gonzalo Pardo Villalibre, autor y desarrollador del *framework* de entrenamiento de redes neuronales orientadas a HAR. Estas reuniones tuvieron como objetivo explicaciones de las diferentes partes del sistema de entrenamiento que ha desarrollado [12] y resolución de dudas.
11. **Diseño y entrenamiento de las redes neuronales:** Tras llevar a cabo la adquisición de conocimientos sobre qué tipo de sistemas y algoritmos de *Machine Learning y Deep*

Learning forman parte del Estado del Arte, es hora de implementarlos y realizar pruebas con diferentes arquitecturas para comprobar si podemos obtener buenos resultados.

1.4. Hardware y Software Empleados

En este apartado del primer capítulo nos encargaremos de concretar y enumerar todos los elementos, tanto de *hardware* como de *software*, utilizados a lo largo de todo el desarrollo de este proyecto.

Los elementos de *hardware* empleados en el desarrollo de este TFG han sido los siguientes:

- Ordenador alojado en la ETSIT de la UVa, formado por los siguientes componentes:
  - **CPU**: Intel Box Core i9 Processor i9-9900KF 3,60Ghz 16M 1,00
  - **GPU**: VGA Gigabyte GeForce® RTX 2080TI 11GB turbo oc
  - **Placa Base**: Gigabyte GA-Z390-MASTER
  - **RAM**: 2xCrucial Ballistix Sport 16GB/3000 BLS16G4D30BESB
- **Sensores inerciales XSENS DOT**: Empleados en diferentes pruebas previas a la grabación de la base de datos de movimientos [10].
- **Sensores inerciales TwynSens**: Empleados en pruebas previas y para la grabación de la base de datos. Hn sido desarrollados en el grupo de investigación GTI de la ETSIT de la Universidad de Valladolid por el doctorando D. Javier González Alonso [4, 5].

Los elementos de *software* empleados en el desarrollo de este TFG han sido los siguientes:

- **Python**: Este ha sido el lenguaje de programación empleado para el tratamiento de los ficheros de datos generados por los sensores inerciales de la base de datos, así como para generar las gráficas y métricas empleadas en la comparativa.
- **Google Collab**: Es la plataforma empleada para la ejecución de los *scripts* (llamados cuadernos) desarrollados en Python mediante la asignación de una máquina remota bajo el soporte de Google.
- **Unity**: Es el motor de juegos empleado para las aplicaciones que graban y reproducen los ficheros de cuaterniones grabados mediante los IMUs.
- **Docker**: Es un proyecto de código abierto basado en el desarrollo de aplicaciones sobre contenedores, de tal manera que se independice dichas aplicaciones y la infraestructura en la que corren. En nuestro caso, se emplea en el framework utilizado para los entrenamientos de las redes clasificadoras [12].

- **Tensorflow**: Es una biblioteca de código abierto empleada para facilitar la programación de programas que resuelven tareas de aprendizaje automático. En nuestro caso, se emplea en el framework utilizado para los entrenamientos de las redes clasificadoras [12].
- **Keras**: Es una librería empleada en la programación de redes neuronales, osea, para *Machine* y *Deep Learning*. Se ha empleado esta librería para la programación de las diferentes arquitecturas de las redes neuronales utilizadas en la clasificación.
- **Overleaf**: Es un editor de LaTeX basado en la nube. Permite la creación, edición e interpretación de proyectos escritos en LaTeX en línea. Esta plataforma se ha empleado para la redacción completa de la memoria.

1.5. Organización de la Memoria

Ahora se hará una descripción de la distribución de los contenidos desarrollados a lo largo de la presente memoria.

En el **capítulo 2** se ha desarrollado una revisión del Estado del Arte en los temas de Reconocimiento de Actividades Humanas, sistemas de Deep Learning y Sensores Inerciales, haciendo un pequeño resumen sobre las técnicas más novedosas que se están empleando de manera exitosa en diferentes ámbitos o con diferentes objetivos a los nuestros y con diferentes tipos de datos, así como los campos de aplicación del HAR que están surgiendo o se encuentran en desarrollo.

En el **capítulo 3** se explica todo el proceso de adquisición de la base de datos propia y pretratamiento de los datos; el cálculo de parámetros cinemáticos, como los ángulos de las partes del cuerpo o de las articulaciones; conceptos básicos sobre inteligencia artificial, Aprendizaje Automático y Deep Learning y las propuestas de redes neuronales empleadas para la clasificación.

En el **capítulo 4** se muestran las similitudes y diferencias entre los resultados obtenidos a nivel de métricas angulares con los sistemas empleados para la grabación de sujetos: sensores inerciales y cámaras de vídeo. Se explica el proceso de obtención de los parámetros que se comparan y los resultado obtenidos de esa comparación.

En el **capítulo 5** exponemos los experimentos llevados a cabo con las redes neuronales y datos recogidos, ventajas y desventajas que observamos en cada situación y los resultados que se obtienen en cada experimento.

Finalmente, en el **capítulo 6** hacemos una recapitulación de todas las ideas y conclusiones a las que hemos llegado con la realización de las pruebas y experimentos y se proponen varias lineas de investigación dentro del tema tratado en el trabajo.

Capítulo 2

Revisión del Estado del Arte

En este capítulo del trabajo se presentarán las técnicas más novedosas empleadas en estudios, artículos e investigaciones de los últimos años. Se tratarán los principales campos de investigación en HAR, partiendo desde un punto de vista más general, hacia algo más específico. En concreto, comenzaremos hablando de los usos más novedosos atribuidos a los IMUs y el reconocimiento de actividades y, finalmente, hablaremos sobre los estudios que realicen experimentos similares a los llevados a cabo en este TFG, que es llevar a cabo HAR empleando sensores inerciales y Machine Learning y Deep Learning en la clasificación.

2.1. Estudios sobre la Adquisición de Movimientos con IMUs

En materia de adquisición de movimientos, los sensores más empleados son los acelerómetros, y los IMUs en menor medida, complementándose en numerosas ocasiones. Esto puede hacernos ver que para la captación de actividades, sobre todo en entornos libres, no clínicos como es nuestro caso, tienen una superioridad importante por la comodidad y versatilidad [13]. En los estudios más recientes, la cantidad de sensores empleados en diversos experimentos mayoritariamente es uno, pero hay numerosos casos con dos y tres sensores. Vemos, de nuevo, que los sistemas propuestos del Estado del Arte que se están desarrollando optan por solventar la desventaja más grande de los IMUs, que es la necesidad de llevarlos puestos, por ello se busca reducir al máximo el número de sensores necesarios para los fines específicos deseados, a pesar de que el precio de este tipo de sensores haya descendido en los últimos tiempos.

Las partes del cuerpo más empleadas en la colocación de los sensores son: mayoritariamente la muñeca [14], debido a la comodidad de tener el sensor en forma de pulsera y a que los patrones de movimiento que se generan con los brazos aparentan ser más representativos de las actividades a realizar; también la cadera, la cintura y el pecho.

En definitiva, vemos que la utilización de los IMUs, y otros sensores similares va estrechamente relacionada al desarrollo e investigación en el campo del HAR. Es por este motivo que se han impulsado también la cantidad de experimentos y publicaciones relativas a estos sensores y

con las previsiones de futuro que tiene el HAR indican que no va a cambiar la situación.

2.2. Estudios sobre HAR

Como se ha mencionado anteriormente, la captación de datos es el paso inicial en el proceso del reconocimiento de actividades. Es por eso que en la mayoría de los casos, los experimentos expuestos en publicaciones llevan de la mano el uso de IMUs. El campo de investigación del HAR ha impulsado el desarrollo y mejora en IMUs, ya que actualmente se están demandando soluciones a problemas HAR en campos de aplicación donde hasta hace incluso cinco años no se consideraba como una meta alcanzable.

Estos campos pueden ser, el de la rehabilitación de pacientes. Más concretamente en sujetos que sufren enfermedades neurológicas [15], obesidad [16], mediciones del índice de masa corporal [17] o desajustes cognitivos [15] entre otros.

Otro de los campos que más está demandando esta tecnología es la del análisis deportivo. Donde se emplea el HAR tanto en la evaluación de lesiones como en la mejora en cuanto a la tecnificación de los deportistas en un ámbito concreto. Algunos artículos muestran análisis sobre el golf [18], la natación [19] o incluso deportes de raqueta [20].

2.3. Estudios sobre HAR Empleando IMUs

Para llevar a cabo HAR, como hemos visto según la gran cantidad de literatura generada, una de las formas más extendidas de llevar a cabo la fase de captación es realizarla empleando IMUs. A la hora de llevar a cabo la clasificación, hay diferentes tipos de algoritmos y técnicas que son de uso muy extendido con diversas características. En los siguientes apartados se expondrán los resultado más relevantes obtenidos.

Estudios Previos Empleando Machine Learning Clásico

Las técnicas de Machine Learning clásico, todavía siguen formando parte del Estado del Arte en lo relativo a clasificación. Como es de esperar, para HAR no podría ser diferente. Este tipo de técnicas se emplea en el reconocimiento de patrones mediante la exposición del sistema a ejemplos o datos. Según [21], es posible conseguir el reconocimiento de actividades recogiendo medidas de 7 sujetos con IMUs y placas de presión llevando a cabo acciones de la vida cotidiana como sentarse, andar o subir las escaleras. Para la fase de clasificación, se emplearon las técnicas *K-Nearest-Neighbours (KNN)*, *Support-Vector-Machine (SVM)* y *Decision-Tree (DT)*. Como método para la evaluación de la efectividad de los clasificadores empleados, se usó la precisión, la sensibilidad y el valor predictivo positivo, con unos valores respectivos de 89.64 %, 94.76 % y 94.41 % para KNN, 87.50 %, 92.61 % y 97.04 % para SVM y 91.79 %, 96.26 % y 95.19 %

para DT. Otro artículo que emplea técnicas de ML es [22], que utiliza los algoritmos de SVM, DT y *Random Forest* (RFo). Se emplea la base de datos *PAMAP2*, la cual cuenta con una elevada dimensionalidad y con diferentes fuentes de datos, como acelerómetros, giroscopios o magnetómetros. Llevando a cabo la clasificación de los datos de entrada se consiguen precisiones de hasta el 99.03 % para RF, 97.5 % para DT y 97.9 % para SVM. La información empleada para estos resultados es el conjunto de los tres tipos de sensores de la base de datos.

Estudios Previos Empleando Deep Learning

Las técnicas de Deep Learning clásico suelen mejorar a los resultados que se obtienen con Machine Learning en problemas que requieran un procesado más complejo y se suelen emplear cuando se tiene disponible una mayor cantidad de muestras o ejemplos para entrenamiento y test, ya que si no, no producen buenos resultados. Durante los últimos años, la tendencia ha sido emplear Redes Neuronales Convolucionales (CNN), Redes Neuronales Recurrentes (RNN) y RNN con *Long Short-Time Memory* (RNN-LSTM).

Existe bastante literatura publicada estos últimos años sobre técnicas de DL para HAR. En [23], se emplea una arquitectura RNN con celdas LSTM y alimentado a la red con la densidad espectral de potencia (*Power Spectral Density o PSD*) de los inputs. Esta arquitectura la nombraron como PSDRNN. Los inputs son aceleraciones triaxiales capturadas con sensores inerciales, y su PSD es calculada mediante la Transformada Corta de Fourier o STFT. En el artículo, se obtienen precisiones de hasta el 96.52 %. Otra investigación, recogida en [24], emplea una base de datos (*WISDM*) con información de 36 sujetos recogida mediante el acelerómetro de un dispositivo móvil, lo cual la hace bastante extensa, y otra de contenido similar (*PUC-Rio*). Empleando una arquitectura DRNN a lo largo de 200 épocas, consigue una precisión del 97 %, superando con creces modelos de redes CNN propuestas en publicaciones de 2018. Finalmente, cabe destacar [25], el cual emplea diferentes arquitecturas de CNN consiguiendo identificar ciertas tareas de la base de datos (*LARa*) de manera muy satisfactoria con más del 90 % de precisión, aunque otras no superan el 50 % en ambas arquitecturas.

Existen numerosos artículos que comparan la efectividad en HAR empleando IMUs y Deep Learning con técnicas de Machine Learning clásico. En todos los casos, se puede observar cómo las técnicas de DL obtienen mejores métricas, aunque la diferencia sea pequeña, respecto a las de ML. Un claro ejemplo es [26], donde se compara el desempeño de las arquitecturas SVM, CNN y LSTM, entre otras. Las bases de datos empleadas son dos: *UCI HAR*, que contiene datos sobre acelerómetros y giroscopio de sensores que vestían 30 sujetos mientras realizaban actividades cotidianas y *Pamap2*, que contiene a mayores información sobre un magnetómetro, temperatura, pulso sobre 9 sujetos. Los valores de las diagonales obtenidos en las matrices de confusión para el primer dataset son similares y los más superiores para la CNN y la SVM, entre 88 % y el 100 %. Para el segundo dataset el comportamiento de la SVM empeora, mientras que el de la LSTM mejora considerablemente. La CNN mantiene su buen comportamiento, mostrando la mayor precisión en ambos datasets. Esta vez empleando un sensores de profundidad, [27] propone una arquitectura que unifica una red 3D-CNN y SVM llegando a tasas de precisión del 97 %. Por último, en [28], utilizando los datos del acelerómetro de un dispositivo móvil, propone una arquitectura de RNN-LSTM ligera y la compara con diferentes algoritmos de ML

clásico (como *Análisis de Componentes Principales, Modelo Oculto de Marokv, Random Forest, Regresión Logística (LR)*) y DL (como LSTM, y CNN) propuestos por diferentes artículos del Estado del Arte, con precisiones del 95.78 %. Los algoritmos de DL consiguieron resultados de precisión ligeramente inferiores, pero aún así se encuentran por encima de todos los sistemas de ML clásico. A pesar de que en el artículo [25], mencionado en el párrafo anterior, no se obtengan de manera general buenos resultados, en los últimos artículos expuestos, se ha demostrado que las redes CNN son una técnica de DL competitiva con las que muestran métricas de calidad más altas.

2.4. Conclusión

Con todo lo revisado en los apartados anteriores, vemos que el método de captura de datos que puede resultar más ventajoso para HAR son los sensores inerciales, debido a la libertad que suponen en entornos libres o no-clínicos. El hecho de tener que llevarlos durante el seguimiento no desvirtúa esta opción por encima de otras como las captura de vídeo o las placas de presión. Por otra parte, el algoritmo de clasificación que se emplee, está claro que depende de la situación y del problema en concreto, pero también está claro que en datasets con poca cantidad de información o una dimensionalidad relativamente baja, los modelos de Deep Learning se encontrarán con una insuficiencia de datos para poder desarrollar todo su potencial. En este caso, la mejor opción puede ser emplear los algoritmos KNN o SVM. En caso contrario, es decir, contando con un dataset suficientemente grande y una capacidad de procesamiento considerable, conviene emplear técnicas de Deep Learning, y las mejores candidatas serían una CNN o una RNN-LSTM. En nuestro caso, no contamos con una gran cantidad de ejemplos para los entrenamientos y test, pero emplearemos técnicas de aumentación de datos o *data agumentation*, explicadas en el capítulo 5 para conseguir muchas más muestras de los sujetos de la base de datos.

Capítulo 3

Materiales y Métodos

3.1. Adquisición de Base de Datos Propia

3.1.1. Motivación para la Grabación de la Base de Datos

Para este proyecto, se decidió no emplear ninguna base de datos pública. El motivo de esta decisión es que en el grupo de investigación al que pertenece el Dr. D. Mario Martínez Zarzuela ya se han desarrollado diferentes proyectos [29, 30, 12] empleando bases de datos de movimientos públicas [31, 32]. Por ello, junto con varios contribuyentes a diferentes proyectos dentro de este grupo de investigación, se llevó a cabo la grabación de nuestra propia base de datos. Además, los movimientos que constituyen la base de datos (Tabla 1) son actividades que se desarrollan en la vida cotidiana y su elección fue recomendada por la Dra. Cristina Simón Martínez.

Junto con las grabaciones de los IMUs, también se grabaron vídeos de los ejercicios para llevar a cabo un trabajo similar al presente para conseguir HAR empleando vídeo [11], del cual se hará una comparativa de los datos obtenidos en ambas grabaciones a lo largo del capítulo 4

Antes de comenzar con la grabación de los sujetos, hicimos diferentes pruebas y grabaciones con los sensores inerciales *Xsens DOT* y los creados por D. Javier González Alonso, llamados *TwynSens* [4, 5]. La idea inicial era hacer las grabaciones empleando ambos tipos de sensores de manera que captaran información similar, situándolos en posiciones corporales parecidas y poder comparar lo obtenido entre ellos. Finalmente, por problemas de conectividad de los sensores comerciales *Xsens DOT*, decidimos descartar su uso y emplear únicamente los *TwynSens* para la confección de la base de datos.

3.1.2. Hardware y Software Empleados

Los *TwynSens* son IMUs que capturan información sobre la orientación de la parte del cuerpo donde están situados en forma de cuaterniones (apartado 3.2.1). Para la grabación de estos datos se empleó una aplicación desarrollada en Unity por el grupo de investigación GTI de la ETS de Ingenieros de Telecomunicación de la Universidad de Valladolid . Esta aplicación permite visualizar en tiempo real los movimientos y demás información que capturan los sensores, así como grabarlo y mostrar los ángulos de las diferentes partes del cuerpo y extremidades. Para la comprobación de las grabaciones, también desarrollaron una aplicación en Unity que sirve de reproductor de las grabaciones registradas con los IMUs en la primera aplicación aquí mencionada.

3.1.3. Protocolo de Grabación

Antes de comenzar con las grabaciones, se presentó toda la documentación requerida por el *Comité de ética de la investigación con medicamentos del área de salud Valladolid este*. También se obtuvieron las autorizaciones pertinentes de todos los sujetos que se ofrecieron para participar en las grabaciones.

Características de los participantes

A continuación, describiremos las características principales sobre los sujetos participantes en la grabación, reclutados mediante los contactos con las personas que forman parte del proyecto de grabación de la base de datos. La edad media de los sujetos estaba en los 21 años. Se realizaron un total de 40 registros, de los cuales 10 personas (25 %) eran mujeres y 30 personas (75 %) eran hombres. El desarrollo del proceso de captura se llevó a cabo en la ETS de Ingenieros de Telecomunicación de la Universidad de Valladolid.

Entre todas las personas involucradas en la grabación de los ejercicios para la conformación de la base de datos, tanto con vídeo como con IMUs, se estableció un protocolo que recoge todas las acciones que se debieron llevar a cabo antes de la llegada de los sujetos, durante las grabaciones y después de las pruebas. El protocolo consiste en:

Antes de la llegada de los sujetos:

- Comprobar la carga de los sensores, aplicar el *heading reset* y tenerlos listos para grabar.
- Comprobar de los formularios de consentimiento.
- Lanzar el programa de grabación de los sensores, el sistema de vídeo, y colocar la cámara en sus marcas.
- Asegurarse de que las marcas estén bien situadas y en buen estado.
- Comprobar los materiales necesarios para la realización de los ejercicios: una silla, una mesa, un bote/botella, ladrillos de construcción de juguete, una pelota y una hoja de papel.

A la llegada de los sujetos:

- Explicar la pruebas: grabación de movimientos con sensores y cámara. Se mostrará un ejemplo en caso de duda.
- Grabar 10 movimientos: 4 pruebas de pierna, 6 pruebas unimanuales y 3 pruebas bimanuales, tanto en el plano sagital como en uno oblicuo a 45º aproximadamente.
- Entregar el guión de las pruebas y consentimiento de los sujetos a la grabación tanto de los movimientos como del vídeo.
- Rellenar una entrada en el registro de grabaciones con los datos demográficos principales del sujeto que llevó a cabo los ejercicios.
- Vestir al sujeto con los sensores. Se emplearán sensores de tipo TwynSens. La posición de los sensores TwynSens es con la entrada microUSB mirando hacia arriba tanto en tren superior como inferior con la cara externa transparente del sensor (o cara superior) hacia afuera del segmento corporal.
- Posteriormente llevar a cabo los 10 ejercicios que se especifican a continuación. En cada ejecución se comprobó la validez de las grabaciones tanto en el formato de cuaterniones como en vídeo.

Durante la grabación:

- Tanto la grabación de sensores como de vídeo comienzan exactamente en el tercer golpe de sincronización realizado de manera manual, mientras que el ejercicio lo hace tras el cuarto.
- El encargado del vídeo (D. Diego Pérez de la Fuente) llevó el *tracking* de las pruebas en todo momento, de tal manera que sólo se darán como válidas las pruebas en las que no se detecte *tracking*, desvanecimientos de la *bounding box*, o que el sujeto salga del plano de grabación.
- El encargado de los sensores (D. Javier González Alonso) debe realizar un *heading reset* siempre antes de colocar los sensores en cada sujeto y cerciorarse de la correcta colocación de los sensores. Los sensores deben ser encendidos en paralelo, alineados con una línea recta como referencia (p.ej. el borde de la mesa) y con el puerto de carga microUSB mirando hacia la pantalla donde está el avatar. Esto es crucial ya que en estos sensores es el momento de encendido el que se toma como Heading Reset. Para el Reset por Software del sistema Unity3D, deberemos colocar el sujeto del mismo modo que vemos al avatar en su estado inicial, con las palmas de las manos apoyadas sobre sus muslos y las piernas levemente separadas. Así, durante el Reset, el sujeto mirará al frente, donde está situada la cámara, pero siguiendo las líneas del suelo
- La colocación del sujeto, de la silla y de la mesa debe seguir las marcas pertinentes.
- Se debe informar al sujeto de:
    - Los pasos deben ser lo más uniformes posibles.
    - La forma de caminar debe ser lo más natural posible.

- La posición de inicio de todas las pruebas debe ser la misma en función de la prueba.
- Los cambios de dirección, giros o movimientos más bruscos deberán suavizarse y hacerse de una manera más cuidadosa para evitar que no haya inconsistencias en las grabaciones.
- Los movimientos bimanuales deben partir y finalizar de la posición inicial.

Los ejercicios a realizar son los descritos en los siguientes puntos (Tabla 1):

- Ejercicios de pierna:

A01. **Caminar hacia delante:** En el plano sagital, el sujeto caminará de frente 3 veces ida y vuelta entre las dos marcas del suelo pertinentes.

A02. **Caminar hacia atrás:** En el plano sagital, el sujeto caminará hacia atrás 3 veces ida y vuelta entre las dos mismas marcas en el suelo de la prueba anterior. Esta actividad consta de 3 repeticiones, contando tanto ida como vuelta.

A03. **Caminar sobre una recta:** A 20º del plano sagital, el sujeto caminará 3 veces ida y vuelta sobre una recta marcada en el suelo. Se tiene que tener especial cuidado en los cambios de sentido.

A04. **Levantarse de la silla:** En plano a 45º. Se requiere la silla. El sujeto se levantará y se volverá a sentar en una silla 5 veces intentando ralentizar el movimiento ligeramente y con los brazos al frente.

- Ejercicios bimanuales:

A05 y A06. **Mover objeto de posición:** En plano a 45º. Se requiere la silla, la mesa y la botella. Este ejercicio se hará una vez con cada mano. El sujeto llevará la botella de una marca a otra encima de la mesa y la devolverá a su posición inicial 5 veces.

A07 y A08. **Mover objeto a la boca:** En plano a 45º. Se requiere la silla, la mesa y la botella. Este ejercicio se hará una vez con cada mano. El sujeto llevará la botella desde la marca de la mesa hasta su boca y devolverla a la posición inicial 5 veces. No se deberá mover o inclinar la cabeza o la espalda.

A09. **Hacer una torre con piezas de construcción:** En un plano a 45º. Se requiere la silla, la mesa y las piezas de construcción. El sujeto cogerá una pieza alternativamente con cada mano y las superpondrá haciendo una torre. Después deshará la torre con la mano contraria a la que puso la última pieza. La torre puede agarrarse al quitar las piezas.

A10. **Lanzar el balón:** En un plano a 45º. Se requiere el balón. Este ejercicio se hará de pie. El sujeto lanzará el balón con ambas manos a una altura un poco superior a la de su cabeza 10 veces. Se procurará empezar y acabar en la misma posición y no hacer movimientos bruscos.

A11 y A12. **Alcanzar un objeto alto:** En plano a 45º. Se requiere un bote colgado del techo. Este ejercicio se hará una vez con cada mano. El sujeto levantará la mano y alcanzará el bote 5 veces.

A13. **Romper un papel:** En plano a 45º. Se requiere una hoja de papel. Partiendo de la posición de reposo, con la hoja en una mano. El sujeto cogerá la hoja de papel con

ID	EJERCICIO	TIPO	PLANO	# REPETICIONES
A01	AndarFrenteYVuelta	Pierna	Sagital	3
A02	AndarHaciaAtrasYVuelta	Pierna	Sagital	3
A03	AndarSobreLinea	Pierna	Oblicuo a $20°$	3
A04	Sentadillas	Pierna	Oblicuo a $45°$	5
A05	MoverVasoDerecha	Unimanual	Oblicuo a $45°$	5
A06	MoverVasoIzquierda	Unimanual	Oblicuo a $45°$	5
A07	BeberVasoDerecha	Unimanual	Oblicuo a $45°$	5
A08	BeberVasoIzquierda	Unimanual	Oblicuo a $45°$	5
A09	MontarLEGO	Bimanual	Oblicuo a $45°$	1
A10	BalonAlAire	Bimanual	Oblicuo a $45°$	10
A11	CogerBotellaAltaDerecha	Unimanual	Oblicuo a $45°$	5
A12	CogerBotellaAltaIzquierda	Unimanual	Oblicuo a $45°$	5
A13	RomperPapelBola	Bimanual	Oblicuo a $45°$	1

Tabla 1: Ejercicios conformantes de la base de datos

ambas manos, la sostendrá con los brazos levantados formando 90º con el tronco y la romperá a la mitad dos veces. Después, hará una bola con el papel y la lanzará con su mano hábil. Hacia el frente, es decir a 45º.

Tras las grabaciones:

- El encargado de los sensores quitará los mismo, al sujeto.
- Se anotará en la sección de comentarios los sucesos anormales o situaciones a recalcar de la realización de las pruebas.
- Se deberá asegurar que todos los vídeos se encuentran en el directorio correspondiente con el etiquetado id.#actividad.mp4 y los archivos de coordenadas id.#actividad.json.
- Se deberá asegurar que todas las grabaciones en crudo de los sensores se encuentran en el directorio correspondiente con el etiquetado adecuado. La nomenclatura de los ficheros de los sensores seguirá el siguiente modelo:

  - RAWTwynSens*FECHA_HORA*.RESET.txt (para el reset, sólo graba en el momento de pulsar el reset, por lo que un mismo archivo de reset será compartido por varias grabaciones RAW)
  - RAWTwynSens*FECHA_HORA*.txt (grabación RAW correspondiente al día FECHA (MM-DD) a las HORA (hh-mm-ss))
  - TwynSens*FECHA_HORA*.txt (grabación procesada correspondiente al día FECHA (MM-DD) a las HORA (hh-mm-ss))

  En el caso de las grabaciones procesadas se generará también un archivo “Reset” pero este podrá ser ignorado por no aportar información útil.

3.1.4. Colocación de los Sensores

En esta sección se concreta la posición de cada sensor durante todas las grabaciones de los ejercicios de la base de datos. Esta colocación se tuvo en cuenta antes de las pruebas llevadas a cabo. También, cabe aclarar que tanto para los ejercicios de tren superior como de tren inferior se empleó la cantidad de 5 sensores colocados alternativamente para grabar cada tren por separado. Es decir, para los ejercicios de tren inferior, de A01 a A04, se colocaron los sensores HIPS, RUL, RLL, LUL y LLL. Para los ejercicios de tren superior, de A05 a A13, se colocaron los sensores BACK, RUA, RLA, LUA y LLA. En la figura 3 podemos ver puntos de tres colores diferentes, cuyo significado es el siguiente:

- Verde: Corresponde con los sensores que registran los ejercicios de brazo. Todos se sitúan en el tren superior.
- Azul: Corresponde con los sensores que usamos en los ejercicios de pierna. Todos ellos están colocados en el tren inferior.
- Rojo: Estos puntos no se han empleado para la grabación de los ejercicios, pero aparecen debido a que existía la posibilidad de haber grabado algunos movimientos con sensores colocados en estos puntos del cuerpo.

Cada par de sensores de las extremidades debe colocarse verticalmente alineado y asegurarse de la correcta sujeción de los mismo al cuerpo del sujeto.

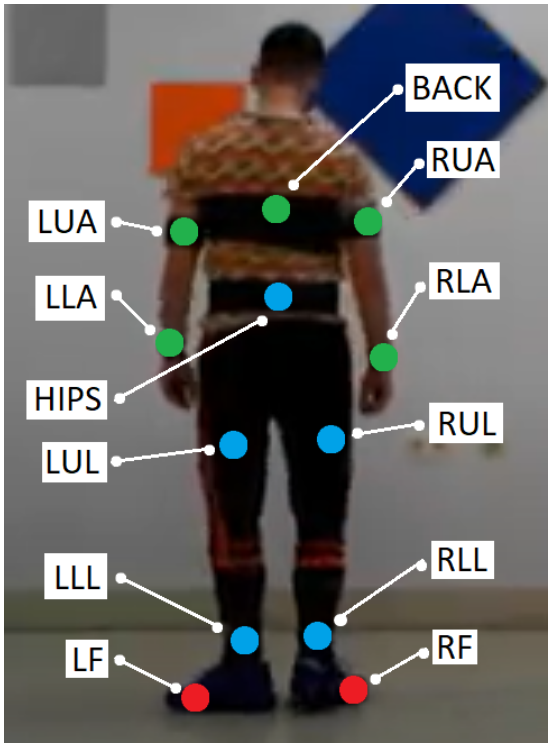

Figura 3: Disposición de los sensores TwynSens

En la tabla 2 podemos ver el significado de las siglas de los nombres de los sensores.

SIGLA	PARTE DEL CUERPO	SIGLA	PARTE DEL CUERPO
BACK	Espacio interescapular	HIPS	Caderas
LUA	Brazo izquierdo	RUA	Brazo derecho
LLA	Antebrazo izquierdo	RLA	Antebrazo derecho
LUL	Muslo izquierdo	RUL	Muslo derecho
LLL	Pantorrilla izquierda	RLL	Pantorrilla derecha
LF	Pie izquierdo	RF	Pie derecho

Tabla 2: Sensores y partes del cuerpo

3.1.5. Preprocesado de las Grabaciones

En este apartado, se expondrá el tratamiento al que se han sometido las grabaciones antes de formar parte de la base de datos. En el apartado 3.2.2 se define cómo son los ejes de referencia que se utilizan para las grabaciones. Para el entendimiento de este apartado no es necesaria su lectura, pero puede ser de ayuda.

Como se especifica en el protocolo de grabación (apartado 3.1.3), al encender los sensores se lleva a cabo un *heading reset*. Esto quiere decir que los sensores fijan la referencia para las rotaciones que van registrar, es decir, los orígenes de los ejes $[x, y, z]$. Este es un punto crítico, ya que requiere que todos los sensores se enciendan con la misma orientación, si no, las medidas que tomaran no representarían los movimientos reales que se han adquirido.

Tras este paso y antes de comenzar con los ejercicios, se debe tomar un *reset* con brazos y piernas estirados (Figura 4), espalda recta y mirando al frente. Es algo similar a una calibración, siempre con la misma orientación, hacia x positivas (Figura 5). Con este reset se registran las rotaciones de los sensores en un instante únicamente. Este *reset* también es un punto crítico. Tras

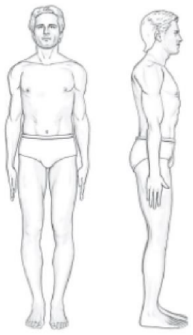

Figura 4: Posición de Reset

la grabación, se obtienen ficheros con las rotaciones de los sensores respecto a su posición del *heading reset*. Esto no es lo que nos interesa, ya que no son representaciones de lo que el sujeto ha hecho realmente. Para conseguir las representaciones reales de los movimientos, es necesario

SUJETO	EJERCICIOS NO DISPONIBLES
S01	-
S02	-
S03	A05-A13
S05	-
S29	-
S30	-
S31	-
S32	-
S33	-
S34	-
S35	-
S36	A07
S37	-
S38	A13
S39	A01-A04
S40	A01-A04

Tabla 3: Contenido Base de Datos definitiva

"restar" la rotación que se ha guardado en el *reset*. Esta transformación se ha aplicado a todas las grabaciones y son los ficheros que conforman la base de datos. La operación en cuestión se corresponde a la ecuación:

$$q_{final} = q_{original} \cdot q_{reset}.inv() \tag{1}$$

Donde $q_{original}$ es cada cuaternión captado con los sensores, $q_{reset}.inv()$ es el cuaternión del reset que representa la rotación inversa y q_{final} es el cuaternión que representa la rotación real desde la posición de reset.

3.1.6. Conformación de la Versión Final de la Base de Datos

Tras la reproducción de las grabaciones en la aplicación desarrollada en Unity que se menciona en 3.1.2 para comprobar su corrección y correspondencia con los ejercicios realizados, se procedió a hacer un filtrado de aquellos que no mostraban corresponderse con lo que debería haberse grabado. Después de inspeccionar los ficheros que mostraban algún tipo de fallo, se vio que en todos ellos alguno de los sensores no transmitió datos durante la grabación de alguno de los ejercicios. La tabla 3 recoge la información de los contenidos de la versión final de la base de datos.

En la base de datos existe un fichero con extensión *.csv* por cada ejercicio e intento de realización. La taxonomía seguida a la hora de nombrar los ficheros ha sido la siguiente: *SXX-AXX-TXX.csv*. Los tres primeros dígitos se corresponden con la identificación del sujeto, los tres siguientes se corresponden con la identificación del ejercicio y los tres últimos dígitos del

nombre se corresponden con el identificador del intento.

3.2. Cálculo de Parámetros Cinemáticos

3.2.1. Introducción a los Cuaterniones

Los cuaterniones son una herramienta algebraica creada por Sir William Rowan Hamilton a principios de la década de 1840. Esta herramienta es una extensión a un espacio tridimensional de los números complejos, es decir, un número complejo con tres componentes en su parte imaginaria. Por lo cual, un cuaternión está formado por cuatro componentes, un escalar o parte real y un vector tridimensional o parte imaginaria, que se define con las componentes unitarias imaginarias $\hat{\imath}, \hat{\jmath}, \hat{k}$ y las relaciones entre estas unidades imaginarias son [33]:

$$\begin{gathered} \hat{\imath}\hat{\jmath} = \hat{k} = -\hat{\jmath}\hat{\imath} \\ \hat{\jmath}\hat{k} = \hat{\imath} = -\hat{k}\hat{\jmath} \\ \hat{k}\hat{\imath} = \hat{\jmath} = -\hat{\imath}\hat{k} \\ \hat{\imath}^2 = \hat{\jmath}^2 = \hat{k}^2 = \hat{\imath}\hat{\jmath}\hat{k} = -1 \end{gathered} \tag{2}$$

Los cuaterniones son una forma compacta y robusta de representar giros o rotaciones en el espacio tridimensional, ya que podríamos expresar una rotación como el paso de una posición, marcada con un vector, a otro. Esto requeriría seis componentes para representar la misma información que con las cuatro de un cuaternión. También se puede representar un giro como una matriz de rotación, la cual tiene dimensión 3x3, lo que harían 9 componentes necesarias para la especificación de la rotación [33, 34, 35, 36].

Los cuaterniones pueden representarse de diferentes formas:

$$\begin{gathered} q := [w, \vec{v}] \qquad w \in \Re, \vec{v} \in \Re^3 \\ q = [w, [x, y, z]] \qquad w, x, y, z \in \Re \\ q = [w, x, y, z] \qquad w, x, y, z \in \Re \\ q = w + x\hat{\imath} + y\hat{\jmath} + z\hat{k} \qquad w, x, y, z \in \Re \end{gathered} \tag{3}$$

También podemos representar puntos y vectores en espacios tridimensionales como cuaterniones, siendo la parte real de los mismo nula y las componentes $[x, y, z]$ las correspondientes al punto o vector a representar:

$$\begin{aligned} P = [P_x, P_y, P_z] \quad &\rightarrow \quad q = [0, P] = [0, P_x, P_y, P_z] \qquad P \in \Re^3 \\ \vec{v} = [v_x, v_y, v_z] \quad &\rightarrow \quad q = [0, \vec{v}] = [0, v_x, v_y, v_z] \qquad \vec{v} \in \Re^3 \end{aligned} \tag{4}$$

Las operaciones definidas para los cuaterniones son la suma:

$$\begin{gathered} q_1 + q_1 = [w_1, \vec{v}_1] + [w_2, \vec{v}_2] \\ K + q = K + [w, \vec{v}] = [K + w, \vec{v}] \qquad K \in \Re \end{gathered} \tag{5}$$

Y el producto:

$$\begin{gathered} q_1 q_2 = [w_1, \vec{v_1}][w_2, \vec{v_2}] = [\vec{v_1} \times \vec{v_2} + w_1\vec{v_2} + w_2\vec{v_1}, w_1 w_2 - \vec{v_1} \cdot \vec{v_2}] \\ K \cdot q = K \cdot [w, \vec{v}] \qquad K \in \Re \end{gathered} \tag{6}$$

Para terminar de describir los conceptos básicos de los cuaterniones definiremos sus operaciones básicas restantes:

Norma o valor absoluto:

$$\begin{gathered} q = [w, x, y, z] \\ \mid q \mid = \sqrt{w^2 + x^2 + y^2 + z^2} \end{gathered} \tag{7}$$

Normalización: Es la definición de un cuaternión asociado al original con una norma unitaria. Un cuaternión no normalizado representa la misma rotación que su normalización. Los cuaterniones unitarios pueden representarse como una esfera 4D, que geométricamente se traduce en una manera muy simple de representación de las rotaciones en un espacio 3D.

$$q' = \frac{q}{\mid q \mid} \tag{8}$$

Conjugado:

$$\begin{gathered} q = [w, x, y, z] \\ q* = [w, -x, -y, -z] \end{gathered} \tag{9}$$

Inverso: Esta no es la representación de la rotación inversa al cuaternión original, pero es el cuaternión que cumple con la definición de elemento inverso de la operación producto. El cuaternión que representa la rotación inversa se calcula mediante la transformación a matriz de rotación.

$$\begin{gathered} q = [w, x, y, z] \\ q^{-1} = \frac{q*}{\mid q \mid^2} \end{gathered} \tag{10}$$

3.2.2. Sistemas de Referencia Coordenados

A continuación se definirán los ejes coordenados que se generan al llevar a cabo el encendido de los sensores. En la figura 5 podemos ver un esquema de la Sala Lorenzo Torres Quevedo de la ETS de Ingenieros de Telecomunicación de la UVa, donde se realizaron las grabaciones para la base de datos. En la figura se muestran los ejes de referencia que se definen de igual manera para todos los sensores empleados.

De esta manera, quedarían definidos: los giros en z serían giros verticales, en el plano XY, los giros en x serían los llevados a cabo en el plano YZ o plano sagital, y los giros en y son los realizados en el plano XZ. Y el convenio de signos empleado sería el de la mano derecha (Figura 6).

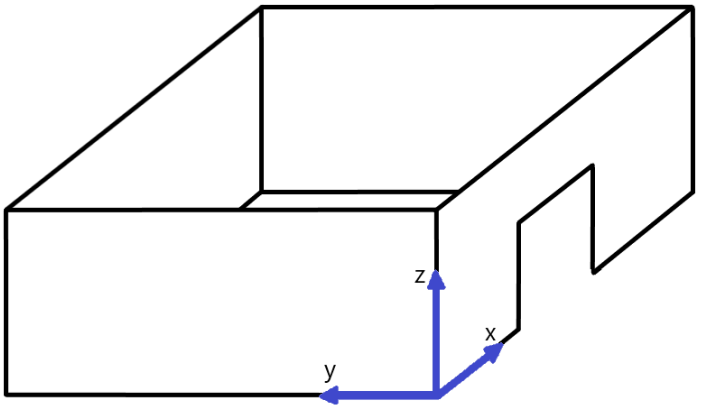

Figura 5: Sistema de referencia Twynsens

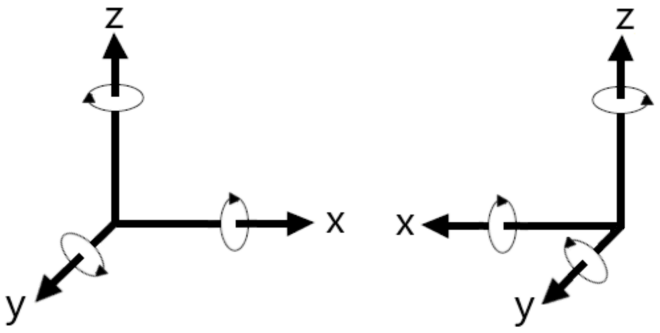

Figura 6: Convenio de signos: Reglas de la mano izquierda y derecha respectivamente

3.2.3. Definición de los Ángulos de Euler: Ángulos de Tait-Bryan

El Teorema de la Rotación de Euler propone que un cuerpo rígido tridimensional, al realizar una rotación, hay al menos un eje del cuerpo que permanece fijo (Figura 7). Gracias a este

teorema, se demostró en 1775 que cualquier rotación tridimensional de un cuerpo sólido podía describirse con 3 componentes y el orden en el que aplicarlas [33].

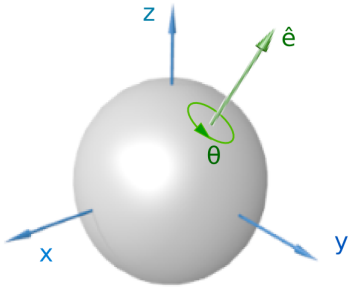

Figura 7: Teorema de rotación de Euler

Estas 3 componentes angulares describen la orientación de un sistema móvil respecto a un sistema de coordenadas fijo. Dependiendo de los ejes que se definan para las tres rotaciones, tenemos dos tipos de rotaciones [37]:

- Rotaciones intrínsecas: Se aplican a los ejes del sistema móvil o rotado (Ej. $[Z, Y, X]$). El sistema de coordenadas de la siguiente rotación es relativo al de la rotación previa.
- Rotaciones extrínsecas: Se aplican a los ejes del sistema fijo (Ej. $[x, y, z]$). El sistema de coordenadas de la siguiente rotación pertenece al sistema de coordenadas fijo.

Una rotación intrínseca es equivale a una rotación extrínseca que emplee el orden de rotación inverso y viceversa, es decir: $[x, y, z] \equiv [Z, Y, X]$.

Los ángulos de Euler se refieren a las tres componentes angulares que se emplean para definir una rotación de un cuerpo tridimensional. Hay diferentes definiciones de estos ángulos, pero las representaciones de los ángulos de Euler más extendidas son los Ángulos Propios de Euler y, sobre todo, los Ángulos de Tait-Bryan (en aeronáutica, y computación gráfica...). Su definición es muy similar, pero los Ángulos de Tait-Bryan son los que hemos empleado en el desarrollo de este trabajo. La principal diferencia entre ambas definiciones es que los Ángulos Propios de Euler emplean rotaciones en únicamente dos ejes (Ej. $[z, x, z]$), mientras que los Ángulos de Tait-Bryan utilizan tres ejes (Ej. $[z, y, x]$) [35].

Una rotación en forma de ángulos de Euler tiene la forma $[\psi, \theta, \varphi]$, pudiendo tomar los valores:

$$\begin{aligned} \psi &\in [-\pi, \pi] \\ \theta &\in [-\pi/2, \pi/2] \\ \varphi &\in [-\pi, \pi] \end{aligned} \tag{11}$$

Para la definición de los anteriores ángulos contamos con:

- Un sistema de referencia fijo $[x, y, z]$.

- Un sistema de referencia móvil $[X, Y, Z]$.
- Un orden de ejes para aplicar las rotaciones.

Partiendo de este escenario, procedemos a llegar hasta la definición de cada componente angular. La intersección entre los planos formados por las dos primeras componentes del orden de los ejes en el sistema de referencia móvil y por las dos últimas componente del orden de los ejes en el sistema de referencia fijo. Por ejemplo, para el orden de ejes $[z, y, x]$ (Figura 8) (una rotación intrínseca) empleamos los planos ZY e yx. Esta intersección se conoce como *N o Línea de Nodos*. Ahora, la componente ψ es el ángulo $\widehat{yN}$, la componente θ es el ángulo $\widehat{xX}$ y el ángulo φ es el ángulo $\widehat{NY}$. El procedimiento sería similar para cualquier otro orden de ejes. En caso de emplear una rotación extrínseca, por definición estaremos obteniendo los ángulos que ha rotado el sólido móvil exactamente en los ejes de referencia [35].

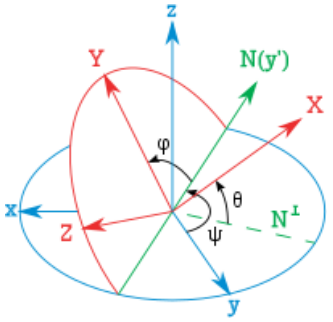

Figura 8: Ejemplo de definición de los ángulos de Tait-Bryan

3.2.4. Cálculo de los Ángulos de Tait-Bryan

Como se ha mencionado con anterioridad en el apartado 3.1.2, la información que obtenemos de los sensores son cuaterniones. En caso de que se requiera una representación más visual y entendible de cara al usuario, es interesante tener una interpretación en ángulos sexagesimales de los cuaterniones registrados. También es interesante comprobar si con esta representación de la información, las métricas de desempeño de las redes neuronales empleadas son mejores o no. Es por estos motivos, que llevaremos a cabo el cálculo de los ángulos que representan los cuaterniones registrados en la base de datos.

Partimos de un cuaternión de la siguiente forma:

$$q = [w, x, y, z] \tag{12}$$

Para hacer la transformación entre cuaternión y ángulo de Euler [38, 39], con componentes angulares (ψ, θ, φ), se emplean fórmulas diferentes según el orden de ejes que utilicemos para emplear las rotaciones [39, 40]. Para el caso de las rotaciones extrínsecas, que es el tipo que

ORDEN DE EJES	PLANO BLOQUEANTE
$[x, y, z]$	y
$[x, z, y]$	z
$[y, x, z]$	x
$[y, z, x]$	z
$[z, x, y]$	x
$[z, y, x]$	y

Tabla 4: Planos productores del bloqueo de Cardán

empleamos en nuestro caso, por ejemplo, para [z x y]:

$$\begin{aligned} \psi &= \arctan(2 \cdot (y \cdot z + w \cdot x), w^2 - x^2 - y^2 + z^2) \\ \theta &= \arcsin(-2 \cdot (x \cdot z - w \cdot y)) \\ \varphi &= \arctan(2 \cdot (x \cdot y + w \cdot z), w^2 + x^2 - y^2 - z^2) \end{aligned} \tag{13}$$

Y teniendo en cuenta que los convenios de signos para los ángulos obtenidos son los explicados en la figura 6. Los ángulos que se representan tienen los nombres: *roll* o cabeceo (eje x), *pitch* o inclinación (eje y) y *yaw* o viraje (eje z).

Gimbal's Lock

En castellano toma el nombre de *bloqueo de Cardán*. Esta es una situación que produce una singularidad al emplear ángulos de Euler. Tomando como referencia la situación de la figura 9, donde vemos una simplificación de un giroscopio, formado por tres ejes rotantes $[x, y, z]$. El bloqueo de Cardán consiste de la pérdida de uno de los grados de libertad que aporta cas uno de esos ejes. Aparece cuando la segunda componente del segundo ángulo de Euler tiene el valor de $\pm\pi/2$ radianes. Es decir, que el eje que produce los bloqueos depende de el orden en el que se apliquen los ángulos de Euler. Esta información está recogida en la tabla 4.

En esta situación los planos que forman el primer y el segundo eje rotante coinciden, haciendo que las rotaciones que apliquemos, sólo produzcan giros con dos grados de libertad [41].

3.2.5. Desplazamiento Angular

Otro parámetro angular que podemos calcular para medir las rotaciones de manera mucho más sencilla es el desplazamiento angular. Según [42, 43], el desplazamiento angular se puede calcular mediante la extensión de la definición del producto interior a cuaterniones.

Para poder aplicar la ecuación, debemos estar trabajando con cuaterniones unitarios, que

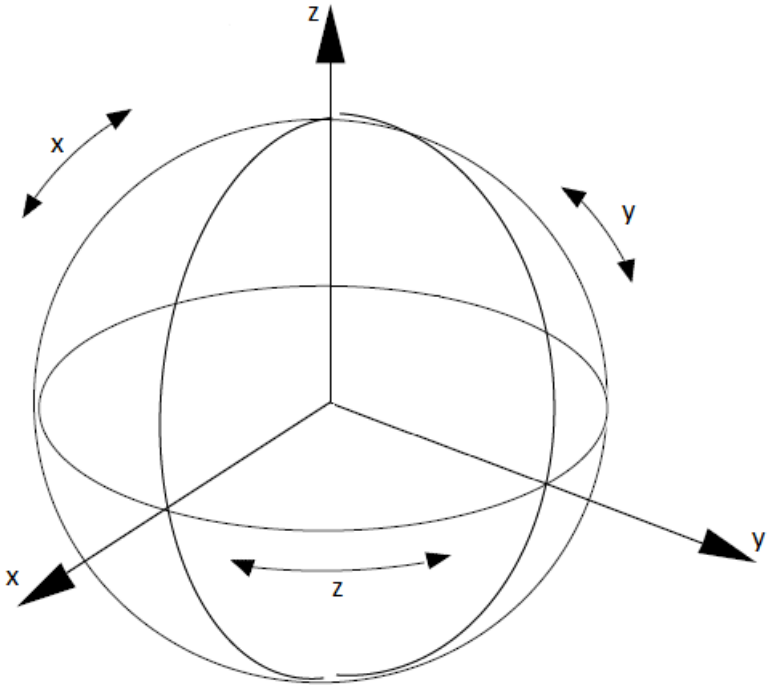

Figura 9: Giroscopio

como estamos midiendo rotaciones mediante sensores, no tenemos preocuparnos por ello, ya que cumplimos con esta condición. También es necesario que para obtener un desplazamiento angular con exactitud los cuaterniones no deben estar muy separados entre sí. Esto tampoco es un problema, ya que la frecuencia de muestreo de los sensores es de 100Hz y no se han llevado a cabo movimientos bruscos o repentinos en las grabaciones. Esto nos asegura la continuidad de las grabaciones y la corta distancia entre cuaterniones consecutivos.

Partiendo de la definición del producto interior:

$$\begin{gathered} q_1 = [w_1, x_1, y_1, z_1] \qquad q_2 = [w_2, x_2, y_2, z_2] \\ \langle q_1 \cdot q_2 \rangle = w_1 \cdot w_2 + x_1 \cdot x_2 + y_1 \cdot y_2 + z_1 \cdot z_2 \\ \cos \frac{\theta}{2} = \langle q_1 \cdot q_2 \rangle \end{gathered} \tag{14}$$

Y despejando la variable angular θ:

$$\theta = 2 \arc\cos[\text{mín}\{abs(q_1 \cdot q_2), 1\}] \tag{15}$$

Es necesario emplear el valor absoluto del producto interior debido a que la ecuación no tiene validez en caso de que el producto sea negativo. También hay que limitar el valor del argumento del $arccos$ a 1, para que no se produzcan ambigüedades en el resultado.

3.2.6. Comparación entre métricas angulares

A continuación, se muestra la figura 10, que representa lo obtenido tras aplicar las operaciones de cálculo de Ángulo de Tait-Brian y Desplazamiento angular al ejercicio caminar hacia delante del sujeto S37, o fichero *S37-A01-T01.csv* como ejemplo cualquiera.

En la gráfica superior, se representan las componentes de los ángulos de Euler $[x, y, z]$ de colores rojo, verde y azul respectivamente. La señal de color rojo representa de manera limpia el ángulo de flexión-extensión la rodilla izquierda, siendo los picos positivos los pasos yendo hacia x positivas y los picos negativos caminando hacia x negativas, como se explica en la figura 5 y la regla de la mano derecha en la figura 6. La señal en verde representa los giros relativos entre los sensores LUL y LLL en el eje y. Esta componente tiene sus principales contribuciones cuando se producen los giros para cambiar de sentido. Y, finalmente, la señal en color azul, muestra los giros relativos en el eje vertical o z entre los sensores LUL y LLL, por ello, en los momentos en los que el sujeto gira, cambia de sentido.

En la gráfica inferior se representa el desplazamiento angular. Como vemos es una magnitud positiva, ya que no tiene en cuenta el criterio de signos de los ejes que establecíamos para los ángulos de Euler. En definitiva, esta magnitud mide la cantidad de rotación relativa entre los sensores LUL y LLL. Es por esto que se observan tanto los picos que se corresponden a la señal roja de la gráfica superior, pero también cobran mucha relevancia las componentes de las señales verde y azul, las cuales se dan en los giros del sujeto. Esto empeora la calidad de la señal al sólo ser relevante las características que se corresponden propiamente al ángulo de flexión-extensión de la rodilla izquierda al caminar.

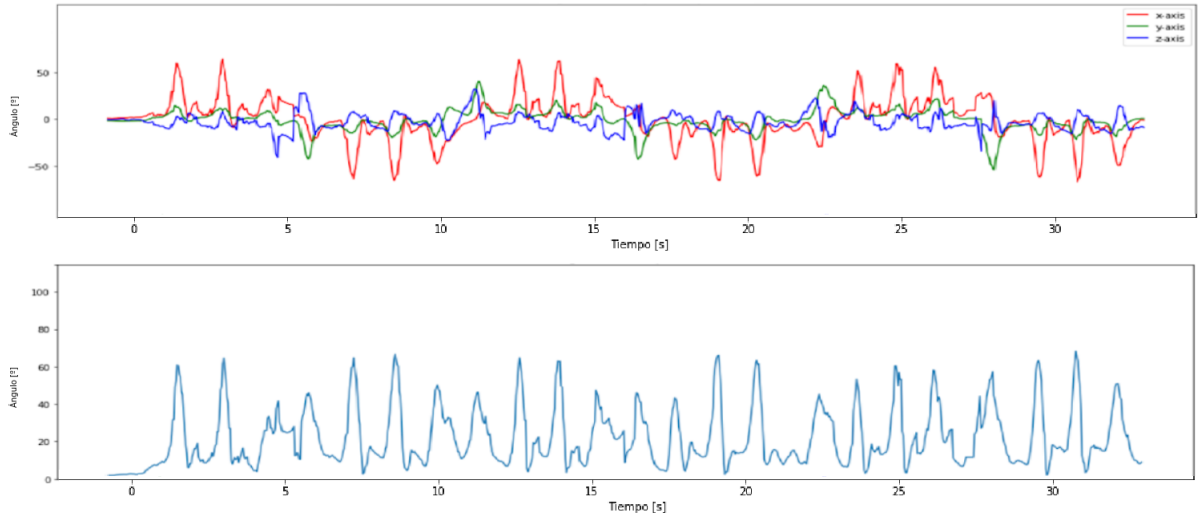

Figura 10: Ángulo de Tait-Brian (superior) y Desplazamiento angular (inferior)

3.3. Redes Neuronales Artificiales

A lo largo de este apartado se tratarán diferentes conceptos básicos sobre inteligencia artificial, aprendizaje automático y aprendizaje profundo, las redes neuronales profundas y técnicas de machine learning empleadas y los métodos para evitar el overfitting o sobreentrenamiento.

Como ayuda visual antes de comenzar con las definiciones, la figura 11 muestra la relación entre inteligencia artificial (AI), Machine Learning (ML) y Deep Learning (DL). En esta figura podemos ver que el ML clásico es un conjunto de técnicas dentro de la AI, y de igual manera, el DL es un conjunto de técnicas dentro del ML clásico. A continuación se explicarán sus definiciones y características.

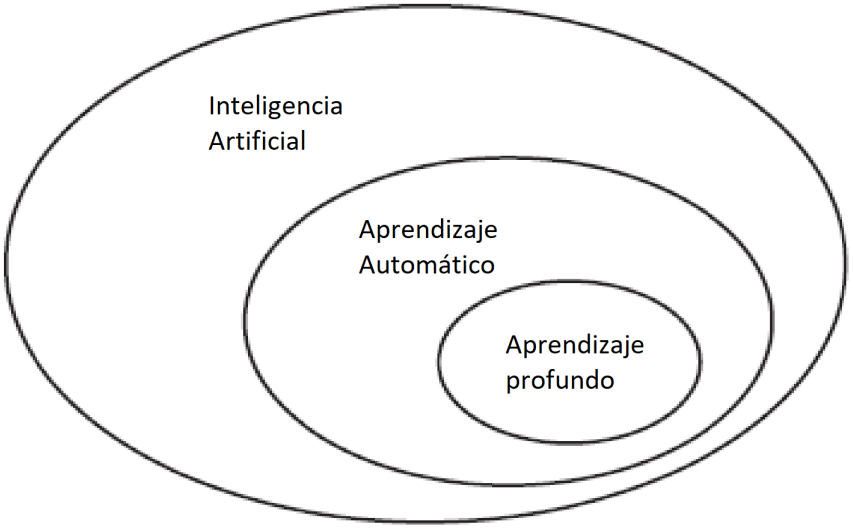

Figura 11: Inteligencia artificial, aprendizaje automático y aprendizaje profundo

3.3.1. Fundamentos de Inteligencia Artificial

El concepto de inteligencia artificial no es completamente cerrado, sino que es bastante flexible y no se considera buena una única definición. Para dar una noción general de lo que se entiende por inteligencia artificial, primero debemos dar una definición, aunque sea genérica de lo que es o se considera la inteligencia.

Según el Diccionario de la Real Academia Española de la Lengua (RAE) [44], la inteligencia se entiende como la *capacidad de entender*, la *capacidad de resolver problemas* o la *habilidad, destreza o experiencia*. Este diccionario también cuenta con una acepción para inteligencia artificial, la cual define por *disciplina científica que se ocupa de crear programas informáticos que ejecutan operaciones comparables a las que realiza la mente humana, como el aprendizaje o el razonamiento lógico*.

En [45] se recogen varias definiciones de AI desde diferentes puntos de vista:

Sistemas que piensan como humanos:

- El excitante nuevo esfuerzo de hacer ordenadores que piensen... Máquinas con mente, en sentido completo y literal (Haugeland, 1985).
- La automatización de actividades que asociamos al pensamiento humano, como toma de decisiones, resolución de problemas, aprender... (Bellman, 1978).

Sistemas que actúan como humanos:

- El arte de crear máquinas que desempeñen funciones que requieran inteligencia cuando las realizan las personas (Kurzweil, 1990).
- El estudio sobre cómo hacer que los ordenadores realicen actividades mejor que la gente (Rich and Knight, 1991).

Sistemas que piensan racionalmente:

- El estudio de las facultades mentales a través de modelos computacionales (Charniak y McDermott, 1985).
- El estudio de la computación que hace posible percibir, razonar y actuar (Winston, 1992).

Sistemas que actúan racionalmente:

- Un campo de estudios que busca explicar y emular comportamientos inteligentes en términos de procesos computacionales (Schalkoff, 1990).
- Rama de la ciencia de ordenadores que se encarga de la automatización del comportamiento inteligente (Luger y Stubblefield, 1993).

Según las definiciones presentadas anteriormente y otras más [46, 47], una máquina inteligente puede ser considerada desde una calculadora hasta un sistema de reconocimiento facial. Con esta última consideración, podemos decir que la inteligencia artificial está constantemente presente en nuestra vida, desde máquinas sencillas, que en un principio no las consideramos como inteligentes, hasta las últimas técnicas de visión artificial.

La IA, al tener una definición tan amplia y abarcar tantos campos y aplicaciones, hay diferentes aproximaciones para los problemas de AI y ésta se divide generalmente en las siguientes ramas:

- **Aprendizaje Automático o Machine Learning:** Es la rama más conocida de la AI. El tema que trata este TFG se encuentra dentro de esta rama, por lo que se desarrollará en el apartado 3.3.2.

- **Métodos probabilísticos:** En esta rama, los sistemas inteligentes llevan a cabo razonamientos con incertidumbre. Esta incertidumbre es en parte por el conocimiento parcial sobre las cosas y es parte de las limitaciones del pensamiento humano.
- **Computación evolutiva:** Las soluciones que se proponen se inspiran en la evolución biológica humana.
- **Teoría del caos:** Son técnicas que se emplean para modelar comportamientos de sistemas que con una pequeña variación en las condiciones iniciales produce una salida muy diferente.
- **Sistemas difusos:** Estos sistemas se basan en que la verdad no es exacta, sino que está definida en una región difusa.

3.3.2. Fundamentos de Aprendizaje Automático

Alrededor de los años 50, el paradigma de programación empleado por programadores para hacer ”máquias inteligentes” había sido la programación simbólica. Este paradigma consiste en la definición explícita de reglas lógicas que el programa seguirá para llevar a cabo sus tareas. Esto funciona muy bien para situaciones en las que el programa trabajará en acciones con esas reglas lógicas muy bien definidas, pero en los casos que se necesiten acciones basadas en respuestas cognitivas, intuiciones o experiencia, este paradigma se queda corto [7].

Esta es la motivación para emplear un nuevo paradigma de programación, el machine learning. El machine learning da la vuelta a la forma de trabajar que tenía la programación simbólica, ya que a partir de unos datos de entrada y las respuestas esperadas para esos datos, el sistema inteligente te genera las reglas que se deben seguir [7]. Es decir, el ML se basa en el reconocimiento de patrones dentro de los datos de entrada del sistema para posteriormente aplicarlo a toma de decisiones, hacer predicciones o incluso minería de datos [48]. Este reconocimiento de patrones en base a grandes cantidades de datos es una solución que en la gran mayoría de casos es hecha a medida.

En definitiva, los sistemas de Machine Learning aplican transformaciones a los datos de entrada para conseguir representaciones más significativas de los mismos. Este proceso se considera ”aprendizaje” y se lleva a cabo mediante la exposición de la red a ejemplos [7]. Su funcionamiento principal es hacer predicciones con cada ejemplo al que es expuesto el sistema y evaluar una función de de coste. Esta función de coste mide el grado de penalización según la predicción diste del resultado esperado. El ejemplo más típico de función de coste puede ser el error cuadrático medio.

Hay tres tipos de sistemas de ML principales:

- **Aprendizaje supervisado:** En este tipo de problemas, los datos que se introducen en el sistema están identificados con una etiqueta. El objetivo del sistema es asignar a cada ejemplo el valor de la etiqueta. En una terminología más técnica, los sistemas supervisados, deben obtener una distribución predictiva $p(t|x)$ para el valor de la etiqueta t, dado un input

x. Dependiendo de la naturaleza de los inputs se pueden distinguir diferentes problemas, por ejemplo, clasificación si las etiquetas toman valores discretos, o regresiones si toman valores continuos.

- **Aprendizaje no supervisado:** Este problema se basa en aprender propiedades de los ejemplos de entrada. De esta manera se puede conseguir *clustering* o agrupamiento de inputs similares, *reducción de dimensionalidad* para representar los datos en un espacio más manejable según conveniencia, etc.
- **Aprendizaje por refuerzo:** Recientemente está recibiendo mucha atención debido a los desarrollos en plataformas de herramientas que apoyan este tipo de métodos de aprendizaje. En el aprendizaje por refuerzo, un agente recibe información sobre el entorno y aprende las acciones necesarias para maximizar las recompensas. Actualmente, los sistemas que emplean este tipo de aprendizaje está en desarrollo e investigación [7].

Para realizar con éxito el correcto entrenamiento de un sistema de Machine Learning, es necesario dividir los datos disponibles en tres grupos:

- **Set de entrenamiento:** Ronda entre el 60-70 % de la totalidad de los datos. Son los inputs de los sistemas y se emplean para hacer que los algoritmos aprendan.
- **Set de validación:** Constituyen aproximadamente el 20 % de los datos disponibles. Este set se emplea para extraer métricas y ajustar los hiperparámetros de los sistemas de ML a lo largo del desarrollo del entrenamiento de los mismos. Estas métricas se pueden extraer cada ciertas iteraciones, al igual que la modificación de los hiperparámetros.
- **Set de test:** Constituye el 20 % restante de las muestras de la base de datos. Se utiliza para comprobar cómo es capaz el modelo entrenado para generalizar, es decir, clasificar datos que no ha visto con anterioridad.

Dependiendo de las técnicas empleadas de ML en la resolución de los problemas, existen diferentes tribus o ramas:

- **Simbolistas:** Esta rama se centra en la lógica. Emplean el paradigma de AI simbólica, explicado anteriormente. Son algoritmos simples bastante rígidos que se basan en reglas de decisión para elegir caminos dentro de su algoritmo. Algunos algoritmos simbolistas pueden ser los *árboles de decisión (DT)* o los *bosques de decisión aleatorios (RFo)*.
- **Conectivistas:** Son algoritmos o sistemas cuya arquitectura se basa en la propia arquitectura del cerebro. Esto quiere decir que las unidades básicas de las arquitecturas se llaman neuronas, que a partir de varias entradas, aplican una función conocida como función de activación y genera una salida. El funcionamiento de estas neuronas simula el de las neuronal cerebrales. Algunos modelos conectivistas son las *Redes Neuronales Artificiales* o las *Redes de Aprendizaje Profundo*. El tema de este TFG se centra en esta rama del Machine Learning, por lo que se desarrollará en el apartado 3.3.3.
- **Evolutistas:** Esta rama toma el ML desde un pensamiento más cercano a la biología, es decir, ver cuál es el crecimiento, las mutaciones y las variaciones de los sistemas, al igual que se puede hacer en cualquier organismo vivo.

- **Bayesianos:** Este acercamiento emplea la estadística. Por ello, las salidas de los sistemas de ML se pueden tomar como una probabilidad, es decir, que todas las salidas posibles tienen cierta probabilidad de aparecer, pero ninguna está determinada. Un algoritmo Bayesiano pueden ser los *modelos de Markov ocultos*.

- **Analogistas:** Los algoritmos analogistas se centran en reconocer similaridades entre las entidades de las diferentes clases que se definen en cada problema. Estos algoritmos son muy empleados en el aprendizaje no supervisado en sistemas del Estado del Arte (apartado 2.3), como el algoritmo *K vecinos más próximos (KNN)* y las *máquinas de vector de soporte (SVM)*.

3.3.3. Fundamentos de Aprendizaje Profundo

A continuación, introduciremos los conceptos teóricos básicos más relevantes sobre Deep Learning para comprender las decisiones y las acciones llevadas a cabo en este trabajo. Se pretenden introducir conceptos generales y necesarios de conocer para el entendimiento de las arquitecturas de red que se emplearán en el presente TFG.

Perceptrón Multicapa

La arquitectura más identificativa y de las más sencillas del DL es el Perceptrón Multicapa (MLP) (Figura 12) o *Feedforward Neural Networks*. Como cualquier algoritmo de DL, su objetivo es hallar una función $f(x)$ que mapee un dato de entrada x a una categoría y, es decir, conseguir clasificarlo. Con el entrenamiento de esta red mediante la exposición de la misma a ejemplos y sus etiquetas, conseguimos definir unos parámetros Θ, llamados pesos, de tal manera que obtenemos $y = f(x; \Theta)$ como mejor función de aproximación a los datos que deseamos clasificar.

El conjunto de la red es una composición de capas y, a su vez, las capas están formadas por neuronas. La cantidad de capas que posea la red es la profundidad de la misma. La primera capa se conoce como *input layer* o capa de entrada, la última como *output layer* o capa de salida y todas las demás capas intermedias son *hidden layers* o capas ocultas, debido a que no se conoce la información que fluye entre ellas a su entrada y su salida.

El motivo por el que este tipo de redes se conozcan como neuronales puede deberse a que su estructura recuerde a la estructura del cerebro humano, donde en diferentes zonas se procesa la información obteniendo conceptos cada vez más complejos o también a que las unidades que conforman las capas, llamadas neuronas, recuerdan al funcionamiento de una neurona cerebral (Figura 13).

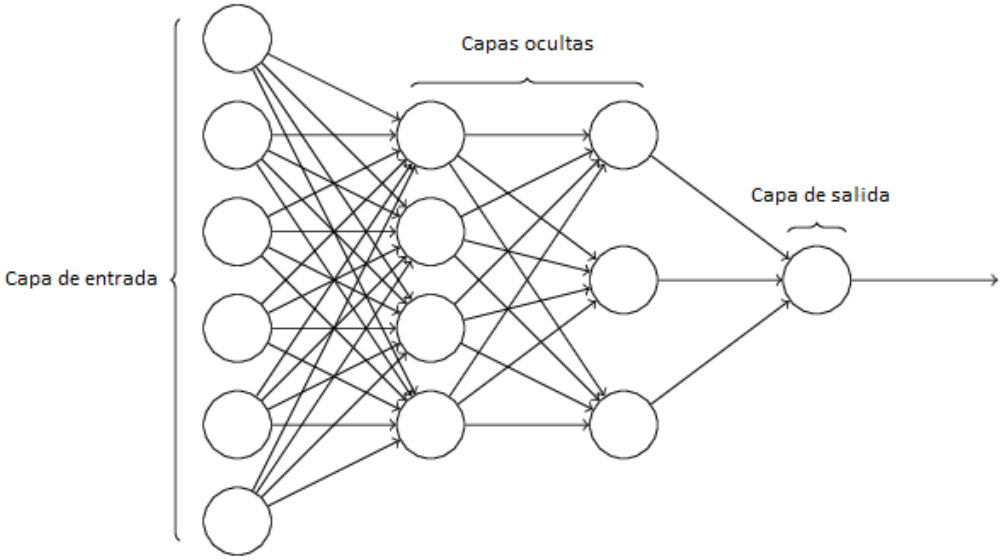

Figura 12: Perceptrón Multicapa

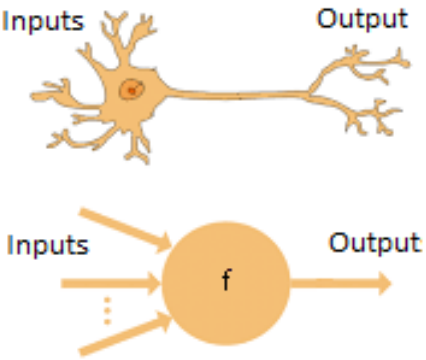

Figura 13: Neurona cerebral (superior) y neurona en Deep Learning (inferior)

Forward Propagation

El nombre con el que se conoce a las redes *feedforward* se refiere al sentido que recorre la información dentro de la red, desde la entrada, a través de todas las capas ocultas y hasta la salida. En la fase de entrenamiento y de test las redes son alimentadas con ejemplos para predecir la clase a la que éstos pertenecen.

El principio de procesamiento de las neuronas en DL de la figura 14 es el siguiente: cada neurona de la capa recibe tantas entradas $[x_1, x_2, ..., x_n]$ como salidas tiene la capa anterior, multiplica a cada entrada por el peso o una ponderación $[w_1, w_2, ..., w_n]$ de la conexión correspondiente y suma los valores, generando el valor $z = x \cdot w^T + b$. Tras esto, se le aplica a z una función de activación $g(z)$, que es la encargada de introducir no-linealidades en las redes. El resultado de la aplicación de estas operaciones es la salida de la neurona $y = g(x \cdot w^T + b)$. La

concatenación de estas operaciones a lo largo de todas las capas de la red, producen en la salida la predicción de la red [49].

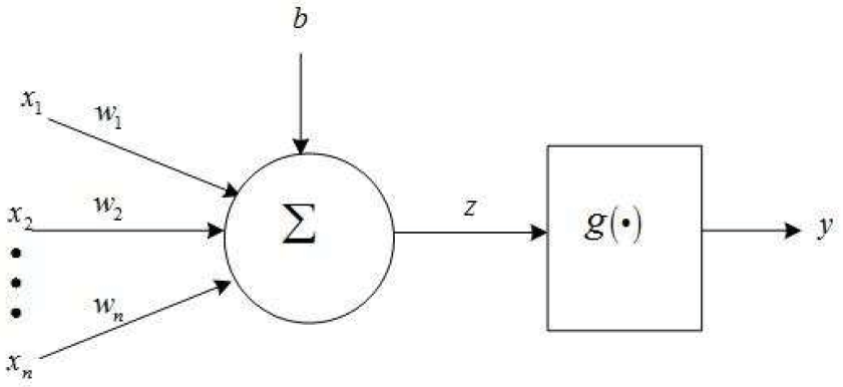

Figura 14: Acciones de la neurona en *forward propagation*

Backpropagation

Las predicciones de las redes tienen parte de error, que se encarga de medir la función de coste o de error $J(\Theta)$, haciendo algo similar a la diferencia entre el valor esperado y el valor predicho. En función del valor de esta función de coste, es necesario cambiar los pesos Θ de forma proporcional a la derivada de la función de error sobre la red para que, en las iteraciones posteriores, la predicción que haga sea más afinada. De esto se encarga el algoritmo backpropagation. Este algoritmo indica la manera en la que se computa el gradiente de la función de coste, el cual es, por lo general, computacionalmente caro. El funcionamiento de back propagation (Figura 15) describe cómo computar $\nabla_\Theta J(\Theta)$, siendo Θ el tensor de los pesos de la red. Llevando a cabo esta operación, se ajustan los pesos de la red y se consigue reducir el valor de la función de coste hasta alcanzar el mínimo global de $J(\Theta)$ [50].

El cómputo de las derivadas parciales (ignorando los efectos de la posible regularización) necesarias para llevar a cabo la disminución de la función de error sería:

$$\frac{\partial J(\Theta)}{\partial \Theta_{ij}^{l}} = a_j^l \cdot \delta_i^{l+1} \tag{16}$$

Siendo $J(\Theta)$ la función de coste de la red, Θ_{ij}^{l} el valor del peso de la neurona ij de la capa l, a_j^l la salida de la función de activación de la neurona j en la capa l y δ_i^{l+1} la diferencia entre el valor predicho por la neurona i de la capa $l+1$ y su valor esperado [3].

Funciones de activación

Estas funciones son operaciones que introducen no-linealidades [49] en las neuronas de las redes, nombradas como $g(z)$ en la figura 14. Las funciones de activación son muy variadas, pero las más empleadas son las siguientes.

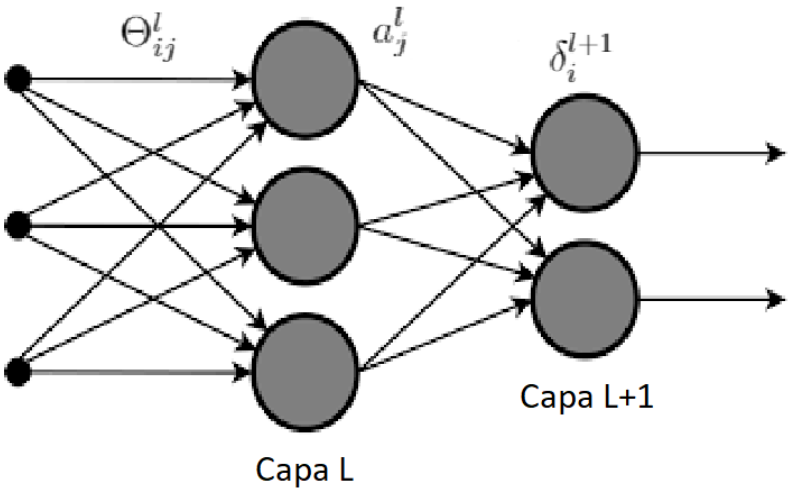

Figura 15: Ejemplo de backprogapation

- **ReLU:** La función unidad lineal rectificada (Figura 16) es una suposición del funcionamiento real de las neuronas cerebrales. Su definición es la siguiente:

$$g(z) = \text{máx}(0, x) \tag{17}$$

Esta función de activación es la más empleada en redes de DL, ya que evita la activación de tantas neuronas como las funciones Sigmoide o Tangente Hiperbólica. Además es computacionalmente más barata y mejora la convergencia de la red. Como la derivada de la ReLU es la función escalón, evita que la red alcance el mínimo local, pero evita el problema del desvanecimiento de gradiente, el cual se explicará posteriormente [49].

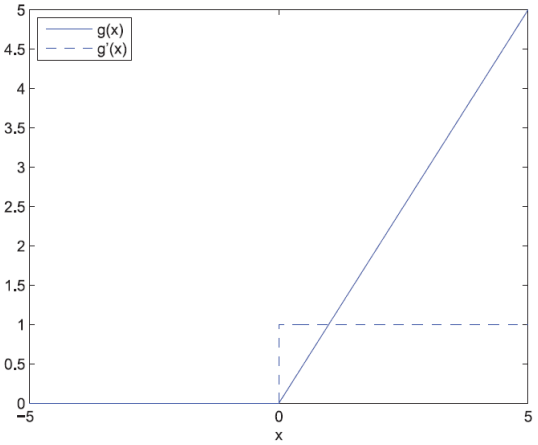

Figura 16: Función de activación ReLU

- **Sigmoide:** La definición de la función Sigmoide es la siguiente:

$$g(z) = \frac{1}{1 + e^{-z}} \tag{18}$$

Esta función tiene la forma que se expone en la figura 17.Su uso tan extendido debido a que su derivada es muy sencilla de calcular, lo que facilita el proceso de *back propagation* que caracteriza a las redes neuronales. Esta función de activación sufre el problema del desvanecimiento del gradiente cuando se emplea en redes con varias capas, aproximadamente en redes con más de cinco capas. Uno de los problemas de la función sigmoide es la rápida saturación que tiene, haciendo muy difícil el entrenamiento de las redes donde es empleada [49].

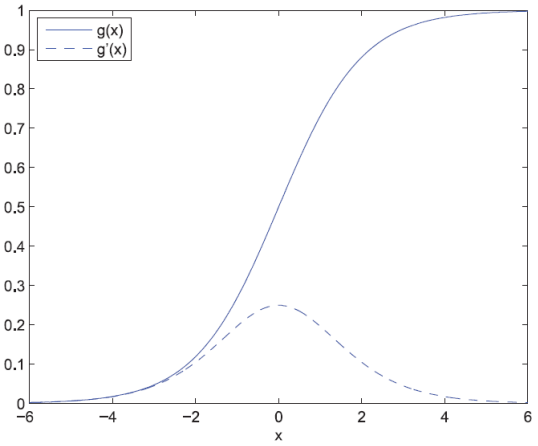

Figura 17: Función de activación Sigmoide

- **Tanh:** La definición de la tangente hiperbólica es la siguiente:

$$g(z) = \frac{\sinh z}{\cosh z} = \frac{e^z - e^{-z}}{e^z + e^{-z}} = 2 \cdot sigmoid(2x) - 1 \tag{19}$$

La diferencia entre las anteriores funciones de activación y esta es que el rango de valores de salida alcanza desde -1 hasta 1 y sus valores de salida son simétricos respecto al origen de coordenadas. Esta simetría hace que la media de los valores de entrada en la capa siguiente esté entorno a cero, lo que les hace tener una ventaja respecto a la función sigmoide. Otra ventaja es que las redes neuronales que emplean la tangente hiperbólica son capaces de alcanzar la convergencia mucho más rápido que las que usan la funciópn sigmoide, consiguiendo a mayores un error de clasificación menor. La contraparte de su uso es que es más compleja computacionalmente y también sufre de desvanecimiento de gradiente [49].

Estas funciones de activación, como hemos visto, tienen sus desventajas, las cuales se han ido solventando con la mejora de los mismos. Se han propuesto modificaciones de estas funciones, como por ejemplo, LReLU, PReLU, RReLU, ELU, MPELU [49].

Desvanecimiento y explosión del gradiente

Estos problemas se dan en los sistemas que emplean redes neuronales con ajuste de los pesos basado en descenso de gradiente y *backpropagation*, que son los sistemas más básicos y los que

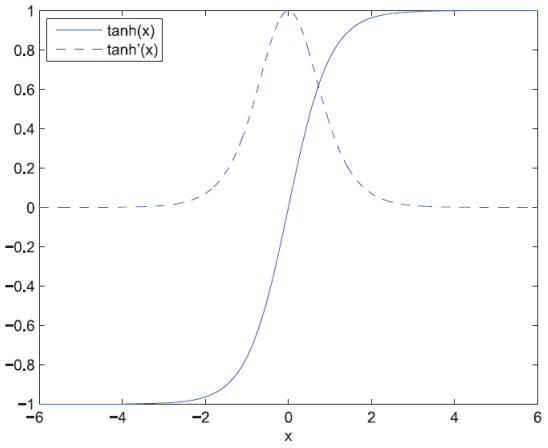

Figura 18: Función de activación Tangente Hiperbólica

se han explicado y empleado en este trabajo. Como se ha explicado en apartados anteriores, la actualización de los pesos se hace proporcionalmente a la derivada parcial de la función de coste $J(\Theta)$.

El caso del desvanecimiento del gradiente, sucede cuando la derivada parcial de la función de coste es demasiado pequeña, evitando una actualización efectiva de los pesos y, en casos extremos, evita que la red aprenda. Este problema también puede afectar en mayor o menos medida según la función de activación que empleemos.

Las funciones de activación empleadas son de vital importancia, ya que la actualización de los pesos es proporcional a su derivada. Según los valores más altos que la derivada alcance, puede llegar a darse el problema de la explosión del gradiente. Esto provoca que las actualizaciones de los pesos son mucho más grandes de lo necesario y, por ende, evita que la red neuronal nunca llegue a converger.

Algunas de las soluciones para evitar sendos problemas relativos al gradiente pueden ser: emplear mejoras de la función de activación ReLU, hacr que la arquitectura de la red no tenga tantas capas, en redes recurrentes, emplear celdas LSTM, regularización de los pesos, etc.

3.3.4. Control del *Overfitting* y del *Underfitting*

El *overfitting* o sobreajuste y el *underfitting* o subajuste (Figura 19) son problemas que pueden ocurrir en todos los sistemas de ML. Se dan cuando hay un desequilibrio entre la optimización y la generalización de las redes. La optimización se refiere a cómo de bien se ajusta el modelo a los datos con los que se ha entrenado la red, mientras que la generalización se refiere a cómo de bien se desenvuelve la red al clasificar datos con los que no se ha entrenado. En los comienzos de los entrenamientos siempre tendremos tasas de pérdida más elevadas tanto en datos de entrenamiento como en datos de test. Esta situación es el momento en el que la red no se ajusta suficientemente a los datos que se pretenden clasificar, es decir, underfitting. Tras varias iteraciones o épocas de entrenamiento, la red consigue a ajustarse bien a los datos,

aumentando la optimización, pero la generalización comienza a empeorar. Esto se debe a que la red empieza a memorizar los ejemplos con los que se ha entrenado, mostrando tasas de pérdida de entrenamiento muy bajas y tasas de pérdida de test elevadas [7].

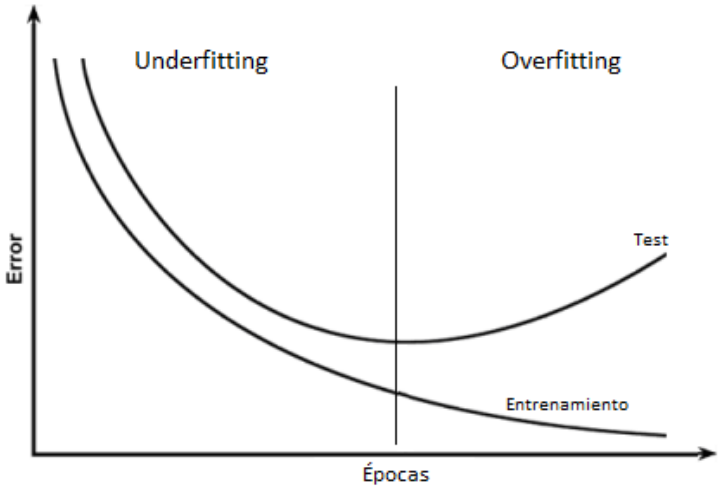

Figura 19: Underfitting y Overfitting

Estos problemas cuentan con diferentes mecanismos para solucionarlos. Mientras el underfitting puede arreglarse con tan solo aumentar el periodo de entrenamiento, número de iteraciones o épocas o la cantidad de datos con la que entrenamos a la red o el tamaño de la red. Por otro lado, el overfitting tiene más variedad y complejidad a la hora de tratarlo.

- **Disminución de la capacidad de la red:** Esta es la solución más sencilla y directa. La red comienza a sufrir overfitting cuando empieza a memorizar los ejemplos con los que se le entrena. Para evitar esto, podemos hacer que a la red le cueste más aprender, y la manera más sencilla de conseguirlo es reduciendo la dimensión de la red y, por consiguiente, la capacidad de la misma.
- **Regularización de los pesos:** La principal idea de la regularización de los pesos es simplificar el modelo que entrenamos con la red neuronal para evitar que suceda el overfitting. Esto se consigue imponiendo restricciones a los pesos de la red para limitar el rango de valores que éstos puedan tomar. La regularización de pesos se implementa añadiendo un término de coste $\Omega(\Theta)$ en la función de pérdidas de la red $\tilde{J}(\Theta; X; y) = J(\Theta; X; y) + \alpha\Omega(\Theta)$, siendo α un hiperparámetro que controla cuánto afecta la regularización a la función de pérdidas. Este término de coste puede definirse de diferentes maneras:

  - **Regularización L^1:** La regularización toma un valor proporcional al valor absoluto de los pesos.

$$\Omega(\Theta) = \frac{1}{2} \mid\mid \omega \mid\mid_1 = \frac{1}{2} \sum_i \mid \omega_i \mid \tag{20}$$

  - **Regularización L^2:** La regularización toma un valor proporcional al cuadrado del valor absoluto de los pesos. A este tipo de regularización también se le conoce como

weight decay o decaimiento de pesos.

$$\Omega(\Theta) = \frac{1}{2} \mid\mid \omega \mid\mid_2^2 \tag{21}$$

- **Early Stop:** Esta técnica se basa en almacenar la información de los hiperparámetros o configuraciones de la red cuando se detecte que el error del set de validación comienza a aumentar durante la duración del entrenamiento y parar el entrenamiento. Al terminar los entrenamientos, se obtienen los parámetros que mejores resultados de error hayan dado, no los últimos empleados. Es una técnica muy utilizada como método de regularización debido a su efectividad y simplicidad [50].
- **Dropout:** Es un algoritmo que mejora notablemente el coste computacional para llevar a cabo una regularización [51]. Esta técnica consiste en la desactivación de múltiples neuronas de las capas ocultas de la red. Este "apagado" de unidades resulta en k subredes generadas a partir de la original 20, cuyo entrenamiento resultará en k modelos, con diferentes métricas de error. Para entrenar estas subredes se emplean mini-lotes del set de entrenamiento original.
- **Normalización del batch:** Es un tipo de capa que normaliza las salidas de una capa de la red de manera adaptativa en función de su media y varianza según evoluciona el entrenamiento. Gracias a ello, conseguimos poder emplear redes mucho más profundas sin alcanzar una situación de overfitting [52].
- **Gradient Clipping:** O *recorte del gradiente* es un método de optimización del gradiente se emplea cuando se tienen en cuenta dependencias de larga duración, como en redes recurrentes que emplean celdas LSTM o GRU. Al emplear funciones de activación con derivadas que pueden tomar tanto valores muy elevados como muy reducidos, las actualizaciones de la función de coste puede tomar valores desmesurados o simplemente no actualizarse. Este algoritmo, consigue que las actualizaciones de la función de coste tomen valores similares, fijando un valor máximo en su actualización.

$$\begin{aligned} Si \mid\mid g \mid\mid > \nu : \\ g' = \frac{g \cdot \nu}{\mid\mid g \mid\mid} \end{aligned} \tag{22}$$

  Siendo g el valor del gradiente, $\mid\mid g \mid\mid$ su norma y ν el valor del umbral.

3.3.5. Redes Neuronales Empleadas

En este apartado del trabajo expondremos las ideas principales detrás de las diferentes arquitecturas de redes neuronales que emplearemos para llevar a cabo el reconocimiento de actividades. En concreto, explicaremos Las redes convolucionales, las recurrentes, y las recurrentes basadas en celdas LSTM.

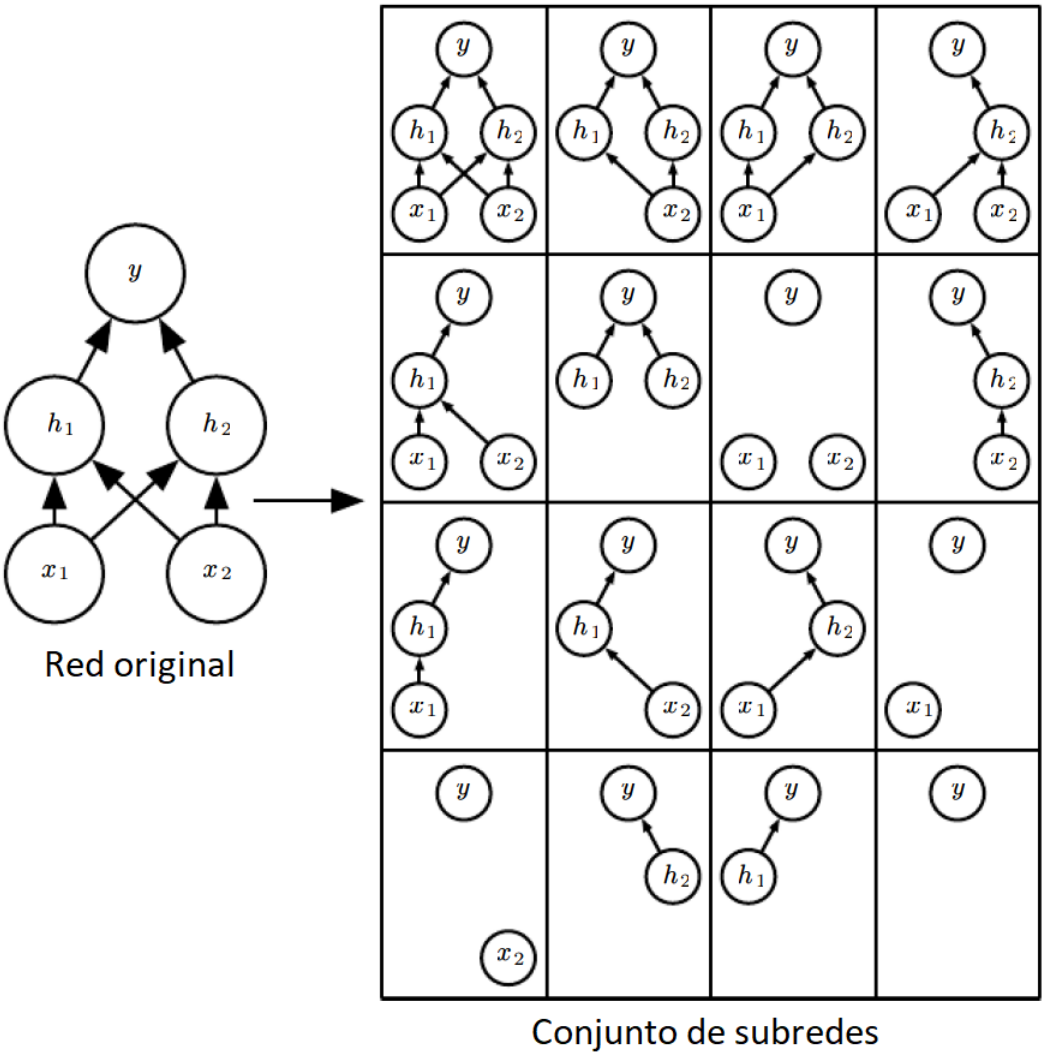

Figura 20: Resultado de la aplicación de dropout

Convolutional Neural Networks o CNN

Las redes neuronales convolucionales han demostrado muy buenos resultados, como se ha visto en la revisión del Estado del Arte, como soluciones propuestas para HAR. Estas redes suelen emplearse cuando los inputs son matrices 2D, como es nuestro caso: en las columnas están situados las diferentes componentes de los IMUs y en las filas se encuentran los instantes temporales. El nombre ”convolucional” se debe a que la operación que realizan las unidades ocultas de sus capas es la convolución, por lo que, una red con al menos una capa de convolución, ya se considera convolucional. Estas capas de convolución están formadas por diferentes capas, como podemos ver en la figura 21 [50].

En la capa convolucional se llevan a cabo convoluciones en paralelo obteniendo diferentes activaciones lineales. Se obtienen diferentes salidas, llamadas mapas de características, aplicando

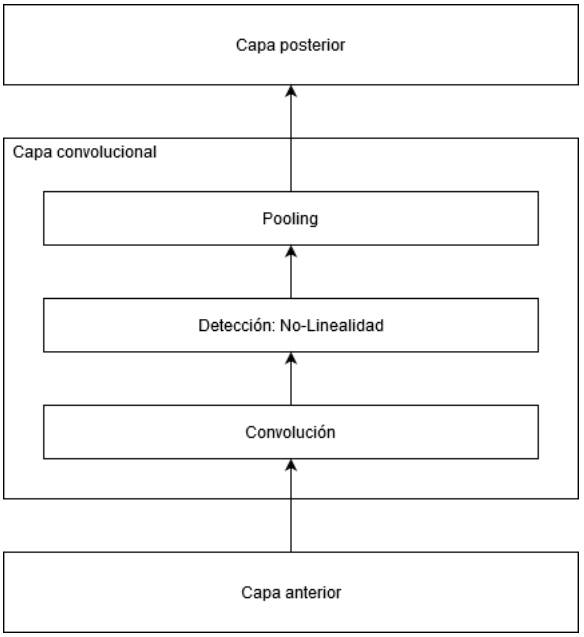

Figura 21: Arquitectura de una capa convolucional

diferentes filtros llamados *kernels* o núcleos de la convolución. Estos mapas de características tienen la propiedad de ser invariantes espacialmente, lo que ayuda a la convergencia de la red. En esta capa pueden modificarse los hiperparámetros de tamaño del kernel, que es la longitud del lado medido en muestras, y el stride, que es el número de muestras que se desplaza el kernel entre operaciones.

En la capa del detector, todos los mapas de características de la capa anterior se someten a funciones de activación a cada uno de estos mapas de características.

Finalmente, en la capa de *pooling* se aplican operaciones que escogen la información más relevante de la capa anterior, como por ejemplo, el valor más alto, la media aritmética o la media ponderada dentro de una ventana. Esta capa de pooling sirve para hacer la salida más invariante a desplazamientos y disminuir notablemente la dimensionalidad de los datos. Proporciona algo similar a un resumen de la salida de las capas anteriores.

Adicionalmente, cada vez es más común añadir capas de normalización del batch, como se explicó en el apartado 3.3.4, permitiendo una mayor profundidad en las redes empleadas [50].

Recurrent Neural Networks o RNN

Las redes neuronales recurrentes son una familia de redes especializadas en el procesado de datos con estructura secuencial, por lo que son ideales para las secuencias temporales. Este tipo de redes permiten escalarse a imágenes con unas dimensiones de largo y ancho muy largas sin añadir complicaciones a los resultados que obtienen.

Una de las ideas principales que soportan la arquitectura es la compartición de parámetros del modelo, permitiendo conseguir una generalización teniendo en cuenta los diferentes instantes temporales de la entrada. En este caso, la aplicación de la operación de convolución se queda como una operación más superficial, sin llegar a conseguir una generalización tan buena como con RNN.

Una forma de representación gráfica de este tipo de redes es mediante gráficos computacionales, que se encargan de describir las operaciones de la red en un estado concreto. Por ello, los gráficos computacionales de las RNN pueden desenrollarse debido a la recurrencia en su arquitectura y comprobar las operaciones que generan la salida de la red. En la figura 22 podemos ver esta idea.

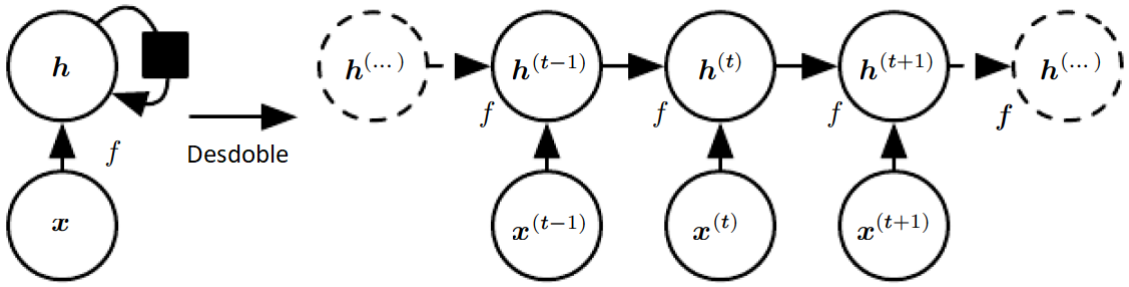

Figura 22: Desenrollado en una RNN

Donde $x^{(t)}$ representa la entrada a la red en un instante temporal t, f es la función de activación de la unidad oculta y $h^{(t)}$ es la hipótesis o predicción de la neurona o unidad oculta.

Long Short-Time Memory o LSTM

En este apartado explicaremos el funcionamiento de las celdas o unidades LSTM. Una celda LSTM son un tipo de unidades ocultas empleadas en redes recurrentes. Su propósito es crear caminos a través del tiempo para relacionar diferentes instantes cuyas derivadas de la función de coste no sufra de los problemas de desvanecimiento o explosión de gradiente.

Este tipo de celdas son capaces de almacenar información sobre instantes temporales pasados, aunque estas unidades son capaces de olvidar los estados anteriores si esa información no es de utilidad para la red. La necesidad de olvidar esta información no es algo que se configure de manera manual, sino que la propia red lo aprende de forma automática. Estas celdas se emplean como sustitución directa de las unidades ocultas típicas en las redes recurrentes.

Como se observa en la figura 23, todas las puertas de la celda están afectadas por una función Sigmoide y no tienen un valor binario, sino que se ven afectadas por pesos propios, lo que es una gran ventaja. Input es la entrada de toda la información a la celda, y la *Puerta Input* controla si la información de *Input* consigue entrar a la celda, evitando la actualización de la propia celda y las dependencias posteriores en la red. La *Puerta Salida* controla la salida de información de la celda. Por último, la *Puerta Olvidar* controla el reseteo del almacenamiento de información de

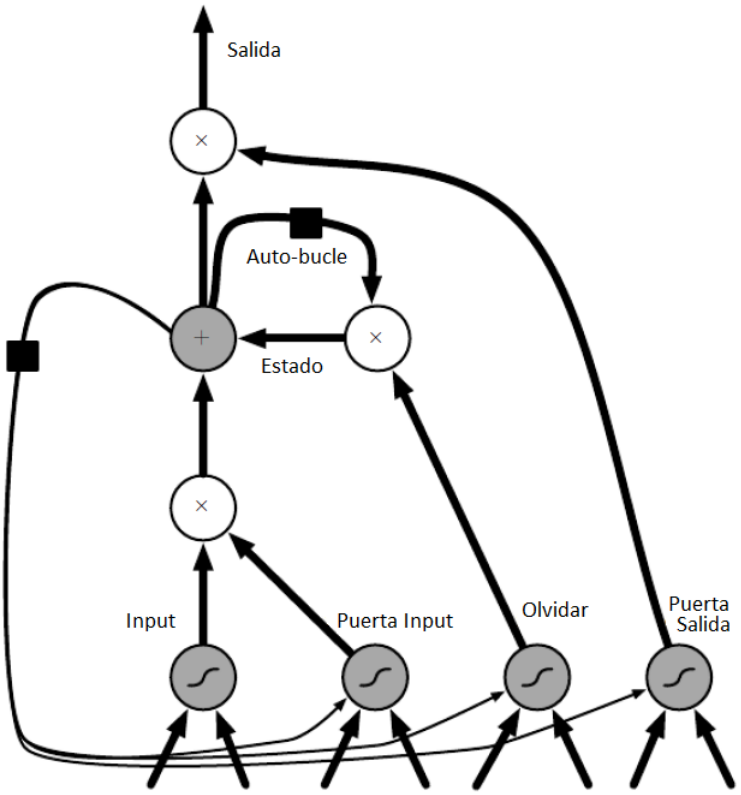

Figura 23: Diagrama de bloques de una celda LSTM

estados anteriores directamente, que se almacena en el *Auto-Bucle*. Los cuadrados de color negro que se ven en la figura 23 corresponden con retardos de un instante de cálculo [50].

Gated Recurrent Unit o GRU

Una celda GRU tiene una función similar a una celda LSTM. La principal diferencia entre estos dos tipos de celdas RNN es que en una GRU tiene únicamente dos puertas: Puerta Actualizar, que hace que el nuevo estado de la celda sustituya al anterior; y Puerta Reset, que provoca que el estado anterior no afecte al nuevo estado de la celda (Figura 24).

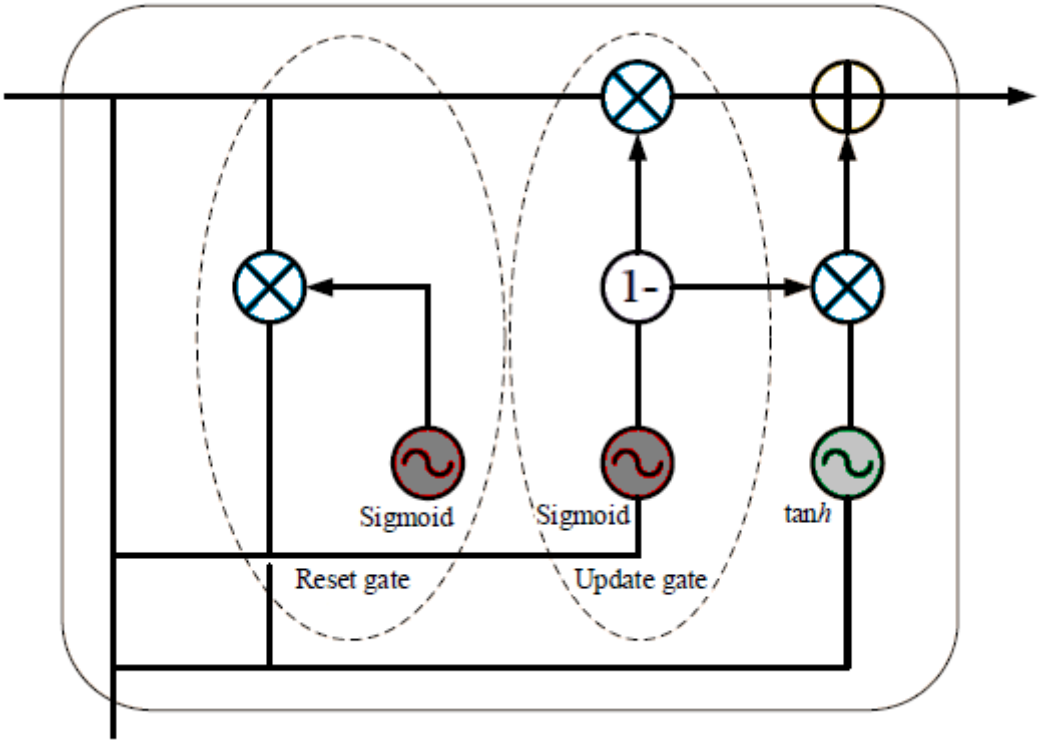

Figura 24: Diagrama de bloques de una celda GRU

Capítulo 4

Comparativa con Sistema de Captura de Datos Mediante Vídeo

En este capítulo del trabajo se pretende hacer una comparativa de la información capturada mediante un sistema de vídeo y el sistema de IMUs explicados en el apartado 3.1.

Durante el desarrollo de este proyecto también se ha desarrollado en paralelo un Trabajo de Fin de Grado con el mismo fin que este: comprobar la capacidad de llevar a cabo HAR, pero empleando vídeo. Este trabajo esta siendo llevado a cabo por *D. Diego Pérez de la Fuente* [11]. A la par que se realizó la grabación de la base de datos de los sensores, también se grabó en vídeo a los sujetos haciendo las actividades definidas en la tabla 1 con la intención de conformar una base de datos similar a la de este trabajo.

En el desarrollo de este apartado, se considera como *Ground-Truth* o verdad absoluta los datos obtenidos por los sensores, pero nos aseguraremos de la congruencia de la información extraída de la base de datos de los IMUs así como de la extraída de la de vídeo. Es importante tener en cuenta a los sistemas de captación de información basados en imagen debido al apoyo que ofrecen a los sistemas basados en sensores vestibles y la relevancia que están ganando en los últimos años gracias a la mejora en sus prestaciones.

La información que se va a comparar serán los ángulos extraídos por ambos sistemas en las actividades A01, A04, A07 y A08 (Tabla 1). Se ha elegido analizar los ángulos de flexión-extensión las rodillas (tanto derecha como izquierda) en los ejercicios de caminar hacia delante (A01) y sentarse o sentadillas (A04), mientras que los ángulos de flexión-extensión de ambos codos y hombros en los ejercicios unimanuales de beber de un vaso (A07 y A08). En estas actividades unimanuales, el codo sufre un ligero movimiento de pronación-supinación entre la posición de reposo y el agarre de la botella, pero éste no es de relevancia para la comparativa. Estas actividades han sido seleccionadas por ser representativas del conjunto de todas las actividades de las bases de datos y para facilitar la visualización gráfica de los resultados.

Pelvis	Meñique pie der.	Muñeca izq.
Cadera izq.	Talón izq.	Muñeca der.
Cadera der.	Talón der.	Nudillo meñique izq.
Torso	Nariz	Nudillo meñique der.
Rodilla izq.	Ojo izq.	Punta corazón izq.
Rodilla der.	Ojo der.	Punta corazón der.
Cuello	Oreja izq.	Nudillo índice izq.
Tobillo izq.	Oreja der.	Nudillo índice der.
Tobillo der.	Hombro izq.	Punta pulgar izq.
Pulgar pie izq.	Hombro der.	Punta ulgar der.
Pulgar pie der.	Codo izq.	
Meñique pie izq.	Codo der.	

Tabla 5: Nombre de los Keypoints

4.1. Motivación

En el campo de *Computer Vision* o Visión Artificial también han sido considerables los avances que se han producido recientemente, y puesto que el HAR es un campo de investigación también complementario a ella, no conviene pasar por alto la gran utilidad que suponen este tipo de herramientas en la fase de captura de movimientos. Teniendo la posibilidad de trabajar en proyectos con convergencia en algunos puntos de la investigación puede servir de apoyo para contrastar los resultados obtenidos por ambos proyectos y la captura de datos es uno de esos puntos de convergencia. Es por esto que resulta conveniente llevar a cabo una comparativa entre los ángulos que se han obtenido con el procesado de la información capturada a través de vídeo con los capturados a través de sensores inerciales. Ambas recogidas de información se han llevado a cabo bajo el mismo protocolo o criterio de captura, descrito en 3.2.5.

4.2. Métodos

4.2.1. Sistema de Captura de Datos Mediante Vídeo

Para llevar a cabo el registro de información mediante vídeo, se ha utilizado una cámara digital de alta resolución y la herramienta *BodyTrack*, disponible dentro del paquete de aplicaciones del proyecto *NVIDIA Maxine: Augmented Reality Software Development Kit (NVIDIA AR SDK)* [53]. Este sistema permite hacer un seguimiento o tracking 3D en tiempo real a partir de vídeo a un sujeto simultáneo. BodyTrack es capaz de asignar hasta 34 keypoints identificando articulaciones o partes del cuerpo concretas recogidas en la tabla 5.

En la figura 25 podemos ver un ejemplo de las zonas del cuerpo donde se asignan los keypoints y los segmentos corporales que identifica BodyTrack, identificando la parte derecha

del cuerpo de color rojo, la izquierda de azul y la cabeza y zona central del tronco de verde.

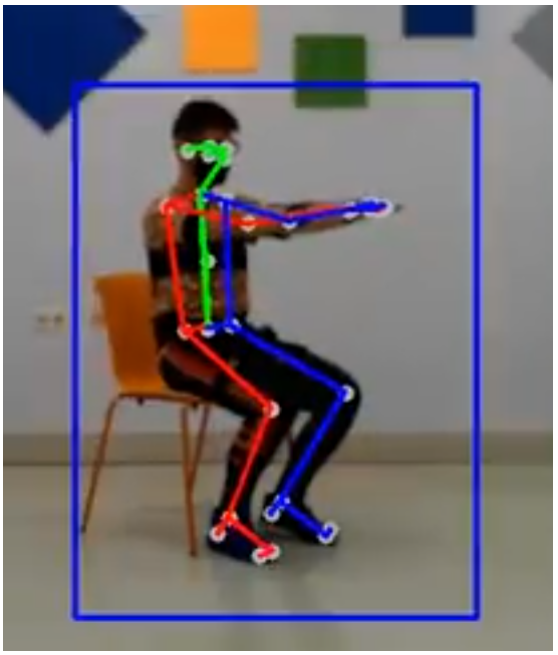

Figura 25: Ejemplo de asignación de Keypoints con BodyTrack

La información obtenida con esta aplicación son las coordenadas $[x, y, z]$ predichas de cada keypoint, siendo el origen de las coordenadas medidas la posición de la cámara. La frecuencia de generación de datos en el fichero de salida por el programa es de 30Hz

4.2.2. Procesado de las Señales

El procesado de las señales es una parte fundamental para poder extraer la información útil de los sistemas de captura de movimientos. En este apartado explicaremos todos los pasos que se han seguido para la extracción directa de información.

Con la información original de vídeo, se ha calculado el ángulo de las articulaciones deseadas mediante el producto escalar. Para poder aplicar el producto escalar, se han tenido que definir los vectores que emplear como diferencia entre las posiciones de dos keypoints. Es decir, para el ángulo del hombro se han empleado los keypoints de la cadera, el del hombro y el del codo; para el ángulo del codo se han empleado el keypoint del hombro, del codo y de la muñeca; y finalmente, para el ángulo de las rodillas se han empleado los keypoints de la cadera, la rodilla y del talón. El software que se ha empleado ha sido *Google Colab*, que permite la ejecución de programas escritos en Python de manera remota en servidores de Google.

El procesado de cada fuente de datos se ha llevado a cabo en diferentes cuadernos de Google Colab, ya que requieren diferente tratamiento, como se muestra en la figura 26.

Para ser capaces de obtener métricas angulares identificativas de cada actividad hemos

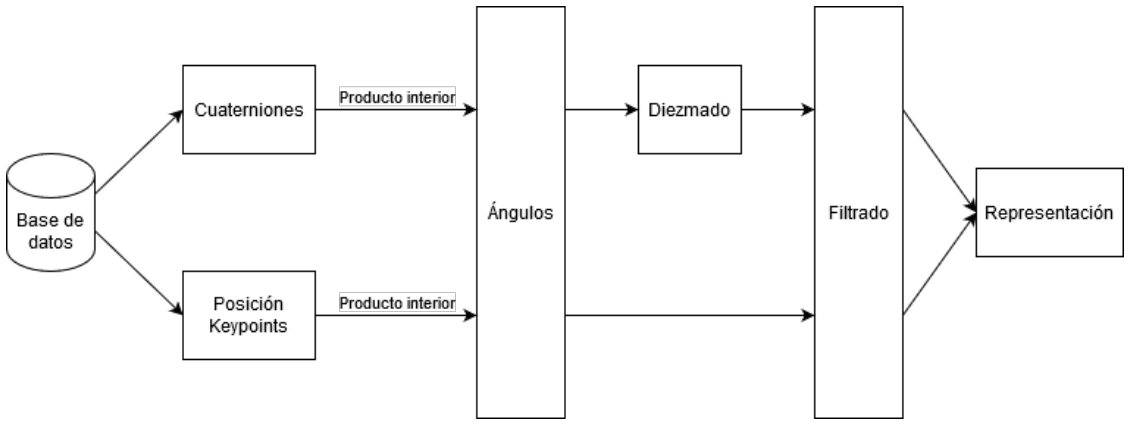

Figura 26: Diagrama de bloques del procesado

aplicado las ecuaciones de los productos interiores adaptadas a la forma de cada tipo de dato. En el procesado de los sensores, comenzamos con la extracción de los cuaterniones, aplicando la ecuación descrita en 3.2.5 para calcular el desplazamiento angular y para el procesado de las posiciones de los keypoints aplicamos la definición de producto interior para vectores de $\Re^3$:

$$\cos\theta = \frac{\vec{u} \cdot \vec{v}}{\mid \vec{u} \mid \cdot \mid \vec{v} \mid} \tag{23}$$

Tras este paso es necesario llevar a cabo un diezmado a las señales de los sensores para igualar sus longitudes a las señales de vídeo y poder llevar a cabo operaciones y cálculos sobre ellas. Esta descompensación de longitudes se debe a que las señales originales han sido adquiridas a diferentes frecuencias, siendo 70Hz la frecuencia de muestreo de los sensores y 30Hz la de vídeo. La tasa de diezmado empleada no ha sido constante en las diferentes actividades analizadas. Esto de debe a la desincronización de las grabaciones. A la hora del comienzo de la grabación, empleamos señales auditivas para sincronizarnos, pero a la hora de finalizar, no fue así. Por ello, ha habido que realizar ajustes en las señales antes del diezmado y en la propia tasa de diezmado.

Y finalmente, se ha sometido a todas las señales a un filtrado para eliminar ruido para las frecuencias más altas y hacer que las señales tengan una forma más suave. Para ello, se ha empleado un filtro de media móvil o *moving average* de longitud de ventana $M = 5$ muestras., cuya ecuación es:

$$y[n] = \frac{1}{M}\sum_{k=0}^{M-1} x[n-k] = \frac{1}{5}\sum_{k=0}^{4} x[n-k] \tag{24}$$

En sensórica es común que haya diferencias de medición. Esto se debe, por lo general, a las diferencias entre métodos de cálculo empleados. Este error de medición se da incluso en dispositivos iguales, lo cual es debido a diferencias o desajuste en la calibración de los dispositivos. En definitiva, para cualquier medida tomada con dos sensores, va a haber un error, por lo que en este caso, empleando diferentes tipos de sistemas de medida, está claro que se van a dar diferencias entre la información obtenida. La métrica que se ha empleado para medir los errores en la medición ha sido la raíz cuadrada del error cuadrático medio o *RMSE*, definido

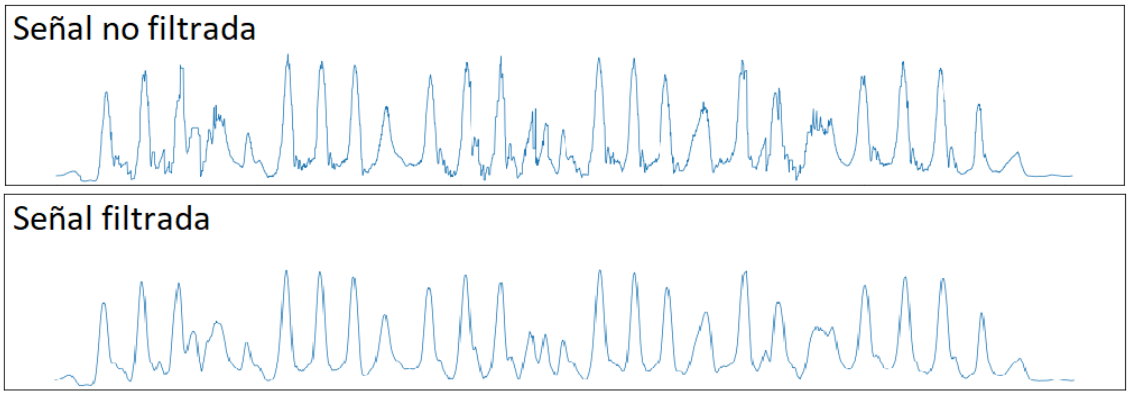

Figura 27: Ejemplo cualitativo del efecto del filtrado

como:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i^{sensor} - y_i^{video})^2} \tag{25}$$

4.3. Comparativa Intra-sujeto

En este apartado de la memoria, se presentan las señales angulares obtenidas mediante los procesos y métodos definidos en los apartados anteriores. En las gráficas que se muestran a continuación podemos observar dos señales, siendo la de color verde la obtenida mediante los IMUs y la azul la obtenida mediante vídeos. En el eje abscisas u horizontal se representa el tiempo, medido en segundos [s]; mientras que en el eje de ordenadas o vertical se representan los ángulos medidos en grados sexagesimales [º] en el rango de 0º a 180º.

Las comparativas se hacen con la información extraída del sujeto llamado S05, viendo las diferencias en cada articulación analizada para las mediciones con IMU y las mediciones con vídeo:

Las figuras 28 y 29 representan los ángulos de las rodillas izquierda y derecha respectivamente adquiridos con los IMUs y con vídeo. Podemos observar seis conjuntos de picos, que se corresponden con las 3 idas y las 3 vueltas del ejercicio A01. También observamos que los picos coinciden con los pasos que el sujeto da en el total de un recorrido, siendo por lo general tres.

Vemos que las señales están perfectamente sincronizadas al inicio y al final de la grabación, produciéndose un pequeño desajuste entre el tercer y quinto conjunto de picos. También, en los periodos de giro, entre conjuntos de picos, se dan las mayores diferencias. Esto se debe a que para los sensores, al medir las diferencias de rotaciones relativas entre dos IMUs, no tienen problema al medir todas las componentes de las rotaciones. Por el contrario, para el sistemas de vídeo, al grabar el ejercicio en un plano sagital respecto a la cámara, tiene mucho más complicado el apreciar las medidas de profundidad, es decir, la componente z, que es la que más relevancia

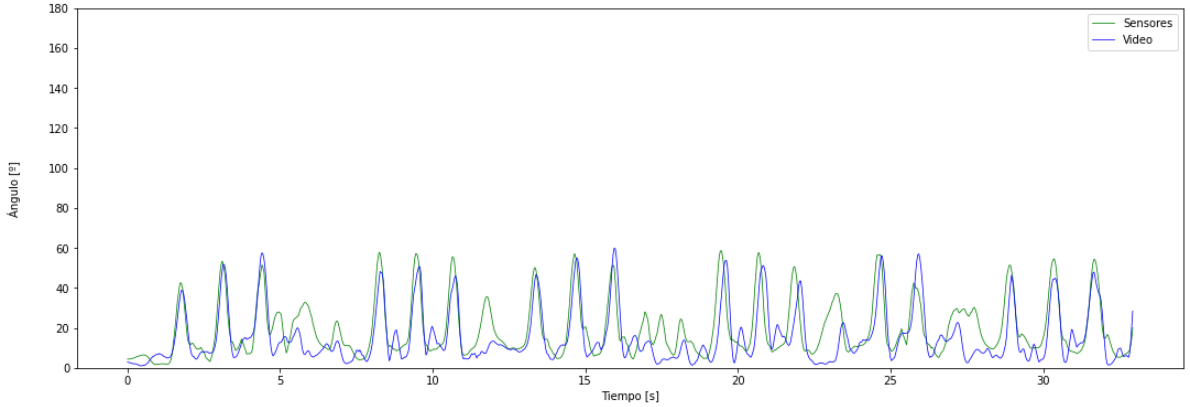

Figura 28: Comparativa S05-A01: AndarFrenteYVuelta - Rodilla izquierda

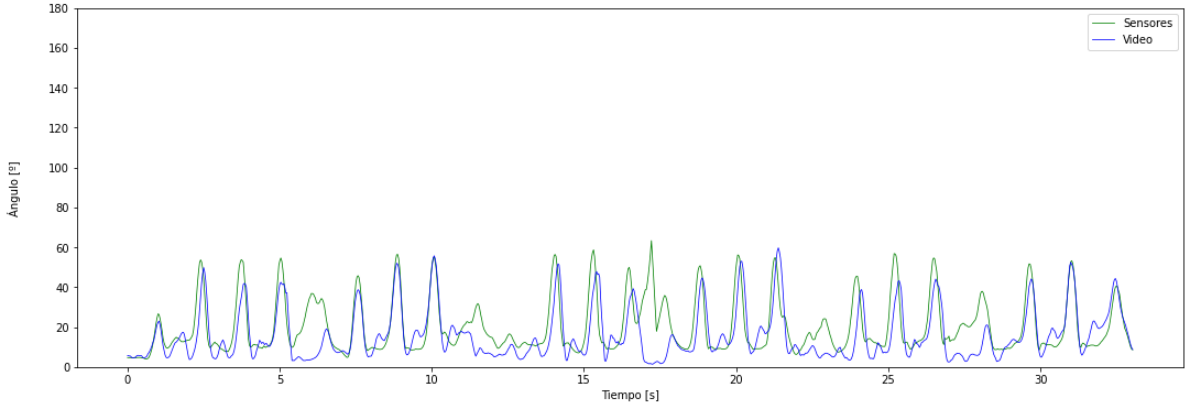

Figura 29: Comparativa S05-A01: AndarFrenteYVuelta - Rodilla derecha

tiene en los giros.

El RMSE para estas figuras ha sido de 10.53 y 11.68 respectivamente, lo cual quiere decir que, por lo general, hay una diferencia de unos 10 entre lo medido con IMUs y lo medido con vídeo. Es observable que las principales diferencias entre las dos señales se dan en los giros, y no así en los ciclos de caminar. Las RMSE tienen un valor similar, lo cual indica que apenas hay diferencias entre las mediciones del lado izquierdo y el lado derecho. Visualmente podemos apreciar que el valor del RMSE puede deberse a los periodos de giro y las desincronizaciones.

Las figuras 30 y 31 representan los ángulos de las rodillas izquierda y derecha respectivamente adquiridos con los IMUs y con vídeo. Podemos observar cinco valles en la forma de onda, que se corresponden con la posición de piernas estiradas (de pie) del ejercicio A04.

En las grabaciones de este ejercicio vemos que la sincronización se mantiene a lo largo de toda la señal, a diferencia del ejercicio anterior. También apreciamos a simple vista que las diferencias entre los sensores y el vídeo es muy superior a la de A01. Este hecho se ve reflejado en las RMSE, con valor de 13.33 y 24.77 respectivamente. Esta diferencia tan grande puede deberse a diferentes motivos:

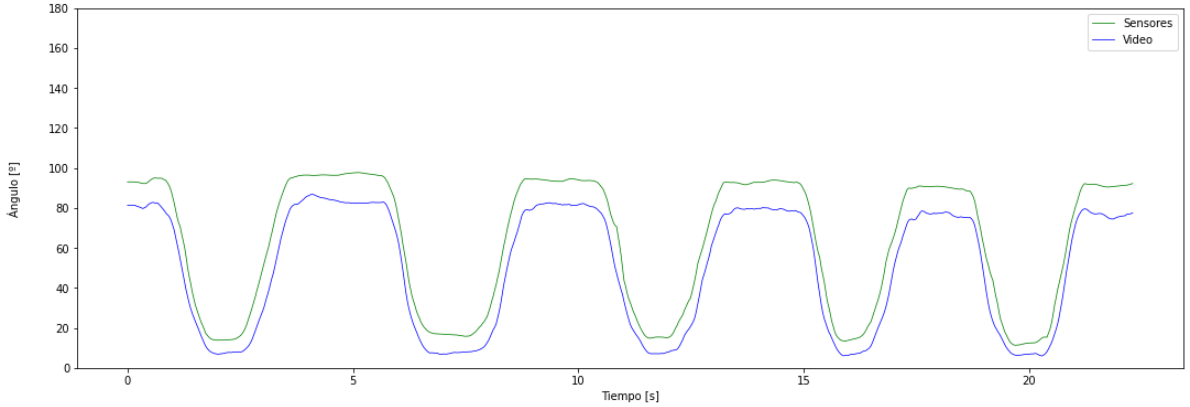

Figura 30: Comparativa S05-A04: Sentadillas - Rodilla izquierda

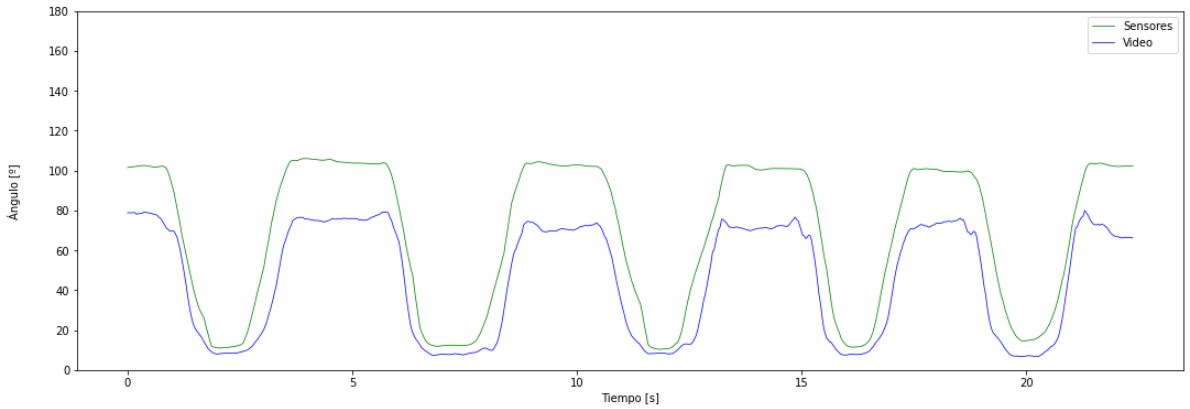

Figura 31: Comparativa S05-A04: Sentadillas - Rodilla derecha

- La diferencia entre las señales de vídeo: Grabación del ejercicio en un plano oblicuo. Como hemos mencionado, el sistema de vídeo, puede tener problemas en la captación de las componentes de profundidad, es por esto que puede haber tanta diferencia tanto entre sensores y vídeo como entre lado izquierdo y derecho.
- La diferencia entre sensores y vídeo: Desviación del Keypoint de la cadera. Para el cálculo de la rodilla, se especificó la utilización de los keypoints de la cadera, de la rodilla y del talón. También, se puede apreciar en la figura 25 que la colocación del keypoint de la cadera en el programa BodyTrack que se hace en la parte más superficial de la zona, no teniendo en cuenta la profundidad real a la que se encuentra la articulación. Esto puede llevar a la reducción del ángulo medido por el sistema de vídeo.

Las figuras 32 y 33 representan los ángulos de los codos izquierdo y derecho respectivamente adquiridos con los IMUs y con vídeo. Podemos observar cuatro de cinco realizaciones completas del ejercicio porque una de las grabaciones se corto unos segundos antes de la finalización del ejercicio.

Entre las dos gráficas, apreciamos la consistencia en las mediciones de los sensores, y no así

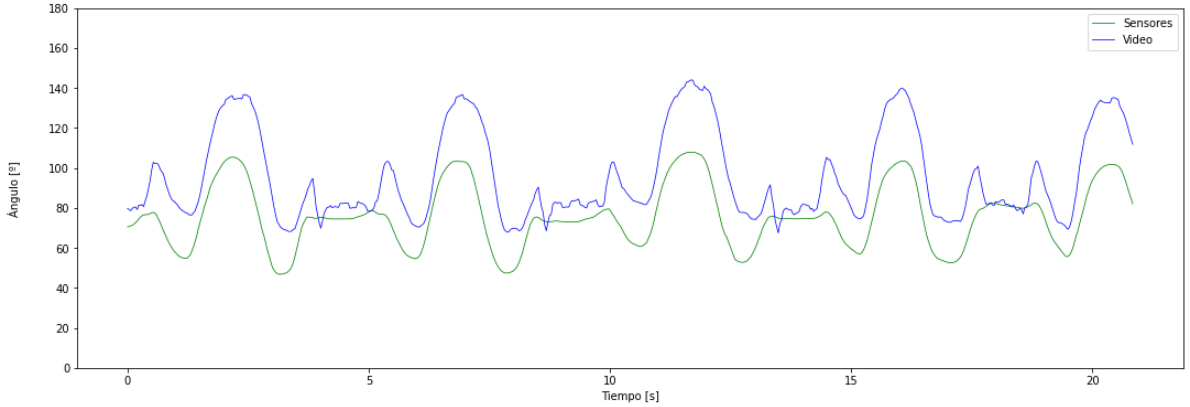

Figura 32: Comparativa S05-A08: BeberVasoIzquierda - Codo izquierdo

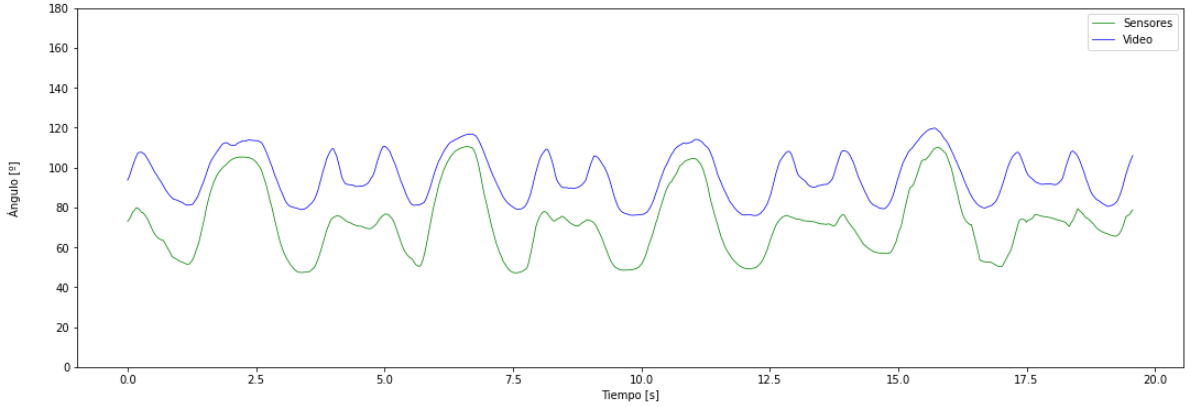

Figura 33: Comparativa S05-A07: BeberVasoDerecha - Codo derecho

en el vídeo. Esto es consistente con el problema de la grabación en un plano oblicuo, ya que el lado izquierdo del cuerpo no estaba al descubierto y el sistema de vídeo es normal que tuviera problemas en la asignación correcta de los keypoints. A mayores de este problema también se sufre algo de desincronización entre las señales en el lado izquierdo. También el problema de la mala estimación de la profundidad por parte del sistema de vídeo hace que las diferencias entre los ángulos calculados sean bastante considerables.

Con estas observaciones tenida en cuenta, los RMSE de 23.41 y 24.55 respectivos, es esperable que tengan un valor tan elevado, pero genera algo de sorpresa que en el lado derecho haya un RMSE mayor. Realmente, tiene sentido, ya que, aunque la grabación de vídeo del lado derecho, pese a tener un parecido mayor con la señal de sensores, las diferencias son mayores que en el lado izquierdo.

Las figuras 34 y 35 representan los ángulos de los hombros izquierdo y derecho respectivamente adquiridos con los IMUs y con vídeo. Podemos observar cuatro de cinco realizaciones completas del ejercicio porque una de las grabaciones se corto unos segundos antes de la finalización del ejercicio.

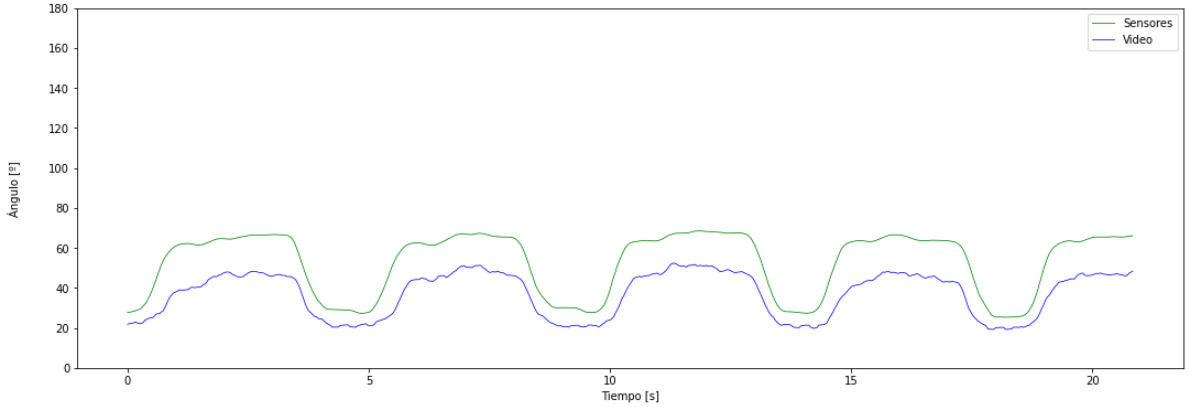

Figura 34: Comparativa S05-A08: BeberVasoIzquierda - Hombro izquierdo

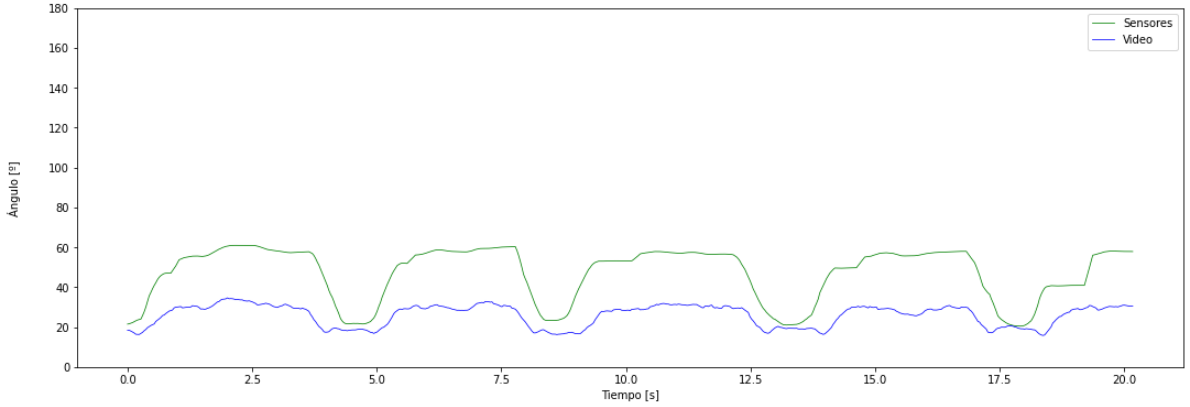

Figura 35: Comparativa S05-A07: BeberVasoDerecha - Hombro derecho

En estas gráficas sigue existiendo una diferencia entre los lados izquierdo y derecho de las grabaciones de vídeo, mientras que los sensores mantienen rangos muy similares. Los problemas de sincronización no son significativos en este caso.

Los RMSE obtenidos para la articulación del hombro son 16.61 y 22.92 respectivamente. SOn valores bastante altos, aunque en el lado izquierdo es ligeramente inferior al resto de ejercicios en general. La diferencia entre ambos RMSE es visualmente apreciable, ya que en las gráficas del hombro izquierdo mantienen una diferencia entre ellas menor que las del hombro derecho.

En resumen, vemos que los IMUs son una herramienta más consistente en las medidas que realiza, dando rangos de valores muy similares y congruentes en los ejercicios examinados. En la adquisición de vídeo, hacen presencia algunas de las desventajas que se mencionan en el capítulo 1, como la dependencia del entorno donde se graba o el ángulo de grabación. Pese a haber diferencias notables entre la información obtenida entre ambos sistemas de adquisición de datos, las capturas con vídeo puede ser igual de válidas para llevar a cabo el reconocimiento de actividades.

EJERCICIO - EXTREMIDAD	S05	S37
A01-RodillaIzquierda	10.53	11.67
A01-RodillaDerecha	11.68	11.19
A04-RodillaIzquierda	13.33	6.13
A04-RodillaDerecha	24.77	7.18
A08-CodoIzquierdo	23.41	17.39
A07-CodoDerecho	24.55	37.85
A08-HombroIzquiero	16.61	11.18
A08-HombroDerecho	22.92	16.02

Tabla 6: RMSE de los sujetos S05 y S37

4.4. Comparativa Inter-sujeto

En este apartado, se presentarán una gráficas similares a las que se encuentran en el apartado anterior, pero esta vez han sido extraídas del sujeto S37. Esta vez se hablará sobre las diferencias entre los valores de la RMSE obtenidas para ambos sujetos (S05 y S37) y ver cómo son los valores entre ambos sujetos a nivel de ángulos obtenidos y de medida del RMSE. La elección estos sujetos se ha basado en la existencia de ambos en las bases de datos tanto de los IMUs como de vídeo, ya que esto no sucede para todos los sujetos.

La tabla 6 contiene un resumen con los cálculos del RMSE realizados sobre los ejercicios analizados en la comparativa. En ella podemos apreciar que los ejercicios de tren inferior tienen un valor del RMSE inferior en comparación a los ejercicios de tren superior. Esta diferencia puede deberse a que los ángulos medidos en los ejercicios de tren inferior tienen un rango de valores menor al que tienen los de tren superior. Podemos ver en la actividad A04 para S05 que el error es comparable al obtenido en los ejercicios de tren superior. Finalmente, podemos decir que estos valores del error son normales y lo más seguro es que se deban a la dificultad de la cámara para captar profundidad, lo cual produce la asignación imprecisa de los keypoints y, por consiguiente, de las mediciones de los ángulos particulares. Como excepción, tenemos la actividad A07 del sujeto S37, donde tenemos un error muy superior, que se discutirá en este apartado.

Las figuras 36 y 37 representan los ángulos de las rodillas izquierda y derecha respectivamente adquiridos con los IMUs y con vídeo. Podemos observar ocho conjuntos de picos, que se corresponden con las tres idas y las tres vueltas del ejercicio A01.

En estas figuras podemos ver que la sincronización entre las señales es bastante malas pero la forma de las ondas es similar a las del S05. Es apreciable que, para los sensores, el valor de los picos es considerablemente inferior, y pueda deberse a que los sensores se hayas desplazado tras el reset.

El valor del RMSE es de 11.67 y 11.19 respectivamente, lo cual es consistente con los valores adquiridos de el sujeto S05. Estos valores tan bajos de error pueden deberse a que el rango

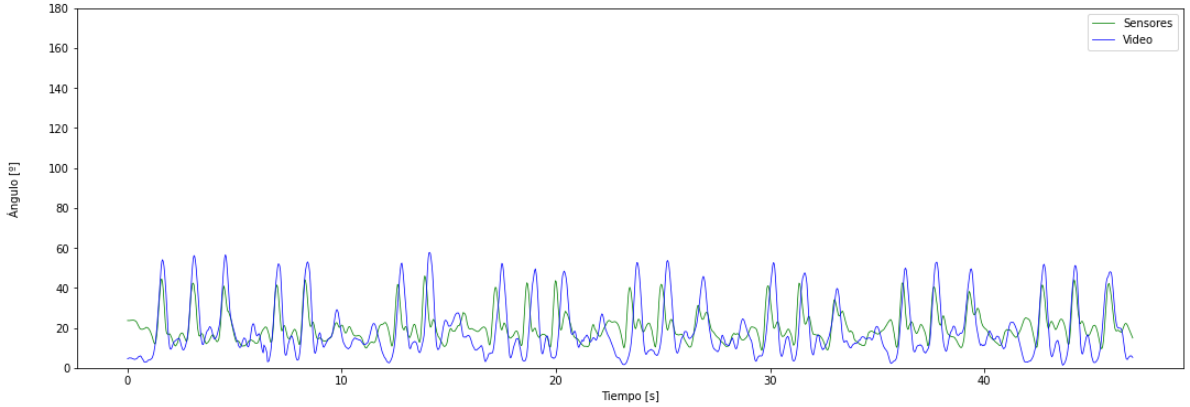

Figura 36: Comparativa S37-A01: AndarFrenteYVuelta - Rodilla izquierda

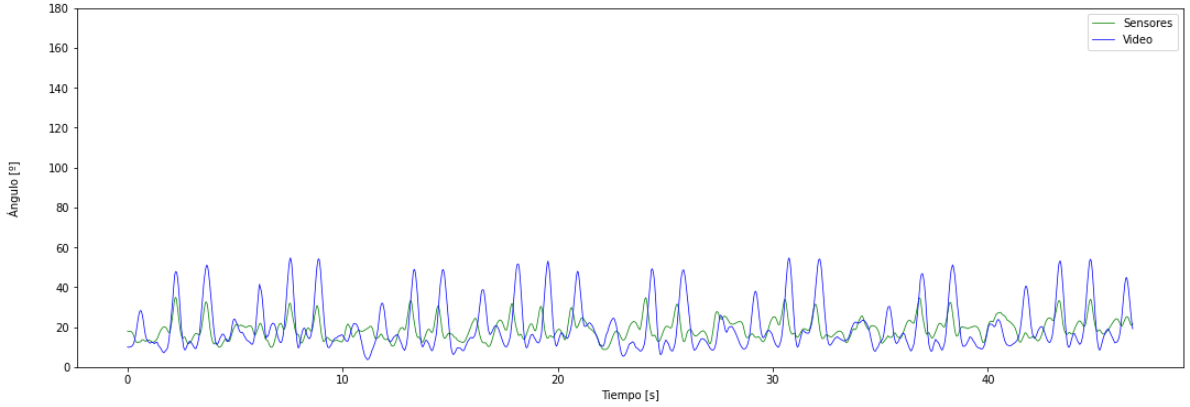

Figura 37: Comparativa S37-A01: AndarFrenteYVuelta - Rodilla derecha

angular de este ejercicio es menor que el rango angular de las demás actividades analizadas en esta comparativa.

Las figuras 38 y 39 representan los ángulos de las rodillas izquierda y derecha respectivamente adquiridos con los IMUs y con vídeo. Podemos observar cuatro valles en la forma de onda, que se corresponden con la posición de piernas estiradas (de pie) del ejercicio A04.

En estas gráficas vemos que los cálculos de los sensores y de la cámara son muy parecidos. No hay problemas de sincronización ni diferencias en los niveles máximos de las señales. Esto contradice la suposición del error en el S05 debido a la colocación del keypoint de la cadera. Esto nos lleva a pensar que la explicación de la diferencia entre los ángulos del S05 sea que la constitución del sujeto a analizar afecta en el sistema de vídeo a la hora de colocación de los keypoints, por lo que el ángulo variará inevitablemente de una persona a otra.

La RMSE obtenida es 6.13 y 7.18 respectivamente. Es algo que visualmente se podía intuir, ya que las señales son muy similares. Es cierto que los máximos de la rodilla derecha son ligeramente menores y más inestables, lo que explica que la RMSE sea algo superior.

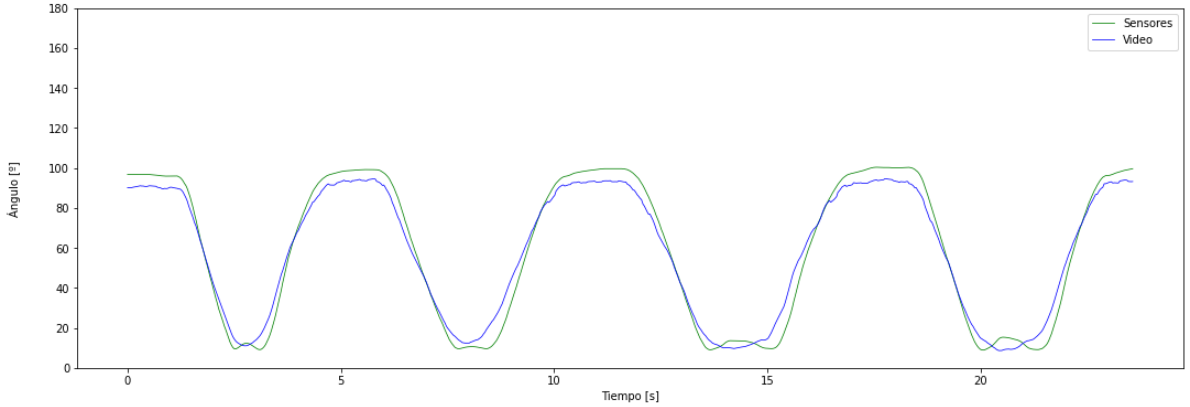

Figura 38: Comparativa S37-A04: Sentadillas - Rodilla izquierda

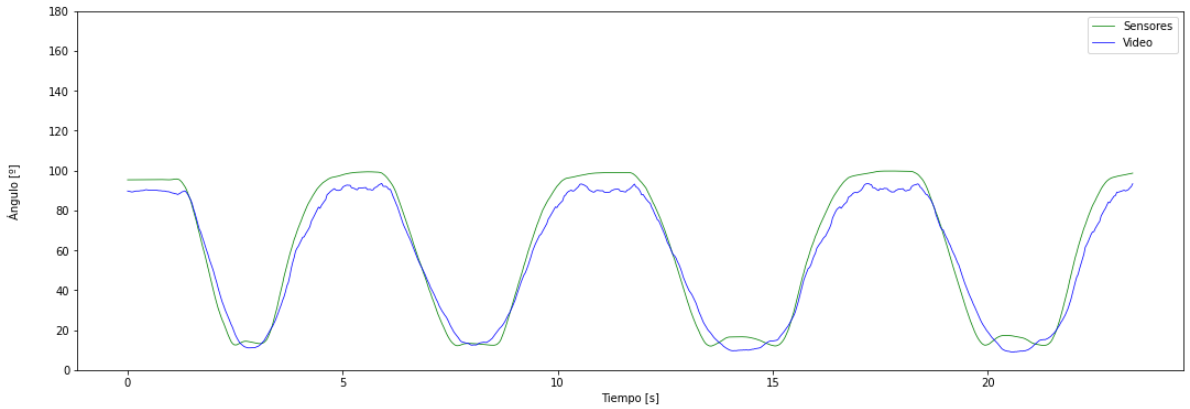

Figura 39: Comparativa S37-A04: Sentadillas - Rodilla derecha

Las figuras 40 y 41 representan los ángulos de los codos izquierdo y derecho respectivamente adquiridos con los IMUs y con vídeo. Podemos observar cinco realizaciones completas del ejercicio.

Para esta realización se observa que la sincronización de la señal de los IMUs empeora considerablemente y la diferencia de valores angulares respecto a lo analizado anteriormente en el lado derecho. Obtenemos un RMSE de 17.39 para el lado izquierdo y de 37.85 para el lado derecho.

El valor del RMSE para el lado izquierdo es bastante alto considerando la escasa diferencia apreciable en la figura 41, pero puede deberse a los picos que se dan antes y después de cada máximo. Estos picos en vídeo pueden exagerarse tanto en comparación con los sensores debido a la perspectiva en la que se sitúa la cámara. Para el lado derecho, ha tenido que darse algún problema con la señal de vídeo, ya que es la que varía de una manera más extrema, a diferencia de la de los IMUs. La forma de onda de los IMUs también varía entre los hemisferios del cuerpo, pero tiene sentido que la manera en la que el sujeto realiza el ejercicio sea diferente entre el brazo izquierdo y el brazo derecho.

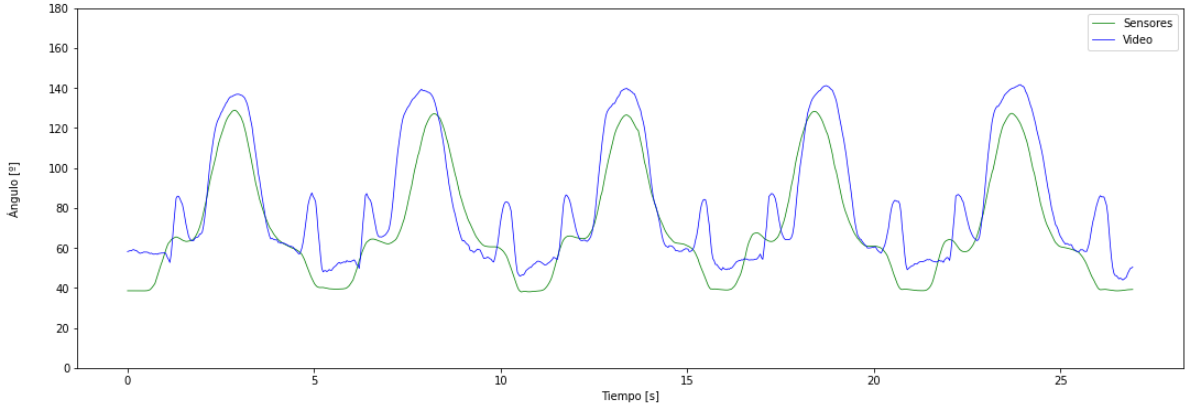

Figura 40: Comparativa S37-A08: BeberVasoIzquierda - Codo izquierdo

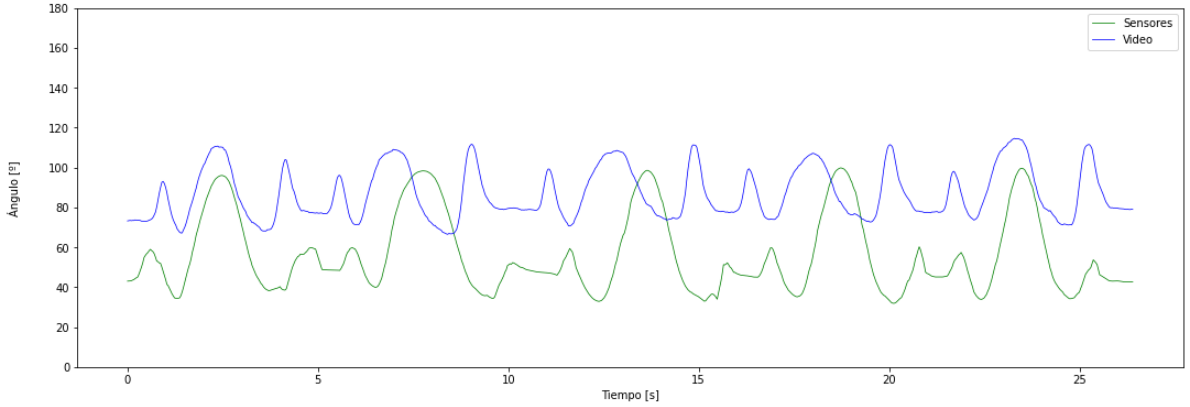

Figura 41: Comparativa S37-A07: BeberVasoDerecha - Codo derecho

Las figuras 42 y 43 representan los ángulos de los hombros izquierdo y derecho respectivamente adquiridos con los IMUs y con vídeo. Podemos observar cinco realizaciones completas del ejercicio.

Para las señales del lado izquierdo no parece que haya problemas en cuanto a sincronización o niveles de medida, con un 11.18 de RMSE. Podemos asumir que el error lo producen, principalmente los picos de la señal de vídeo. Por otro lado, en el ejercicio con el brazo derecho vemos que la forma de onda de la señal de vídeo es muy diferente a la anterior, lo que nos hace sospechar que puede haber habido algún problema en la captura de información. Para el lado derecho obtenemos una RMSE de 16.02, superior a la anterior, lo cual tiene sentido.

En la figura 42 se puede apreciar que la señal de vídeo distingue las partes de agarrar la botella y de llevarla a la boca, mientras los sensores obtienen un ángulo constante. Para la figura 43 podemos decir que los errores en la señal de vídeo han sido cometidos por el ángulo de grabación y la mala apreciación de la componente de profundidad del sistema de vídeo.

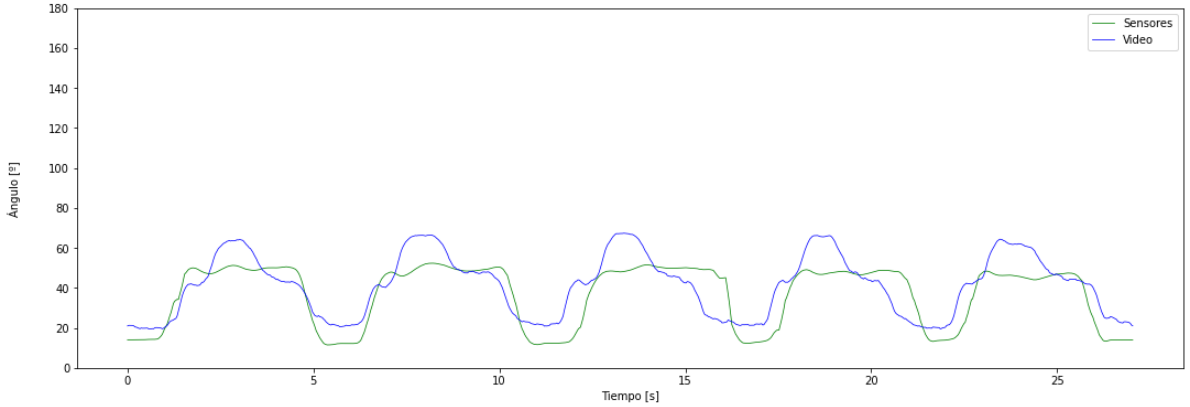

Figura 42: Comparativa S37-A08: BeberVasoIzquierda - Hombro izquierdo

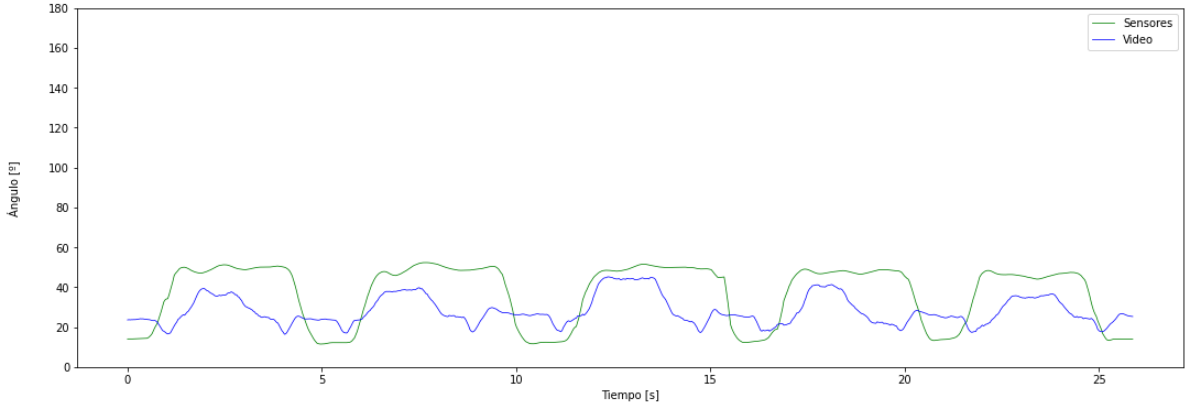

Figura 43: Comparativa S37-A07: BeberVasoDerecha - Hombro derecho

4.5. Conclusión

Las ideas principales obtenidas tras este análisis comparando lo obtenido entre diferentes sujetos son las siguientes:

En los ejercicio grabados en un plano oblicuo respecto de la cámara, se aprecia la carencia del sistema de vídeo a la hora de identificar las componentes de profundidad, produciendo errores en la predicción de los keypoint y, por consiguiente, en los ángulos calculados a partir de ellos. Por el contrario, los sensores inerciales no tienen este problema, ya que el cálculo que realizan es la rotación relativa entre los cuaterniones asociados a dos sensores.

Se observa que la variación en las medidas realizadas por los sensores son más consistentes variando la actividad y el sujeto, pero son muy susceptibles, obviamente, al movimiento de los sensores tras el paso de reset. Por otro lado, esto no es problema para el vídeo. La recogida de datos llevada a cabo para la base de datos de vídeo ha sido mucho más productiva, debido a que, para los sensores se emplea mucho tiempo en la colocación y preparación de los mismos para cada sujeto.

Finalmente, me gustaría resaltar que, en los ejercicios A01, A02 y A03, las medidas obtenidas del RMSE se ven afectadas negativamente en los periodos en los que los sujetos giran para el cambio de dirección. Según lo expuesto en párrafos anteriores, el sistema de vídeo no es capaz de reconocer los ángulos de manera precisa debido a la inconsistencia con la profundidad, por lo que son instantes en los que las medidas pueden llegar a falsear el desempeño de los sistemas a la hora de capturar los ángulos de los ejercicios.

Capítulo 5

Clasificación de Actividades con Aprendizaje Profundo

En este capítulo se explicarán los diferentes experimentos, comprobaciones y pruebas que se han llevado a cabo y los resultados que se han obtenido en cada uno de ellos.

5.1. Framework de entrenamiento

Para el proceso de entrenamiento y evaluación de las redes neuronales que hemos usado, se ha empleado el framework desarrollado por *D. Gonzalo Pardo Villalibre* en su Trabajo de Fin de Grado [12]. Este sistema es una aplicación basada en Docker y su objetivo es la simplificación del proceso de entrenamiento para redes de aprendizaje profundo destinadas a realizar HAR. Este sistema divide el proceso del entrenamiento en 3 entornos: de preprocesado, de entrenamiento y de inferencia. De estos entornos sólo se han empleado los de preprocesado y entrenamiento.

Entorno de preprocesado

Este entorno se encarga de la transformación de los datos con un formato concreto predeterminado y llevar a cabo *data augmentation* o aumento de datos, de tal manera que consiga mejorar el desempeño de la red que utilicemos. Este entorno realiza varias operaciones sobre los datos de entrada.

El framework es capaz de tratar tanto vectores de posición (de 3 componentes) o de orientación o cuaterniones (de 4 componentes). En nuestro caso, al emplear cuaterniones, podemos emplear este framework.

Es necesario que estos datos de entrada estén dispuestos bajo una arquitectura concreta: Un fichero por ejercicio nombrado como *Sujeto-NombreActividad-Intento.csv*, lo cual concuerda

con el formato que tienen los ficheros de la base de datos final que hemos generado, ya que lo hicimos pensando en el futuro uso de este framework. El contenido de los ficheros son matrices bidimensionales en los que sus columnas son las componentes de los cuaterniones de cada sensor y sus filas son los instantes temporales. La nomenclatura de los sensores debe ser *Nombre_0, Nombre_1, Nombre_2 y Nombre_3*. Estos son los únicos requisitos que deben cumplir los ficheros de entrada.

En este entorno se pueden especificar diferentes parámetros de una forma muy compacta, mediante un único fichero de configuración. Algunos de estos parámetros son: la lista de ejercicios que se van a emplear, la lista de sujetos, si los datos son orientaciones o posiciones, el tamaño de la ventana que deseamos, las muestras del *overlap*, la lista de ángulos que deseamos rotar o si deseamos emplear la FFT.

Este entorno está formado por diferentes módulos, encargados de funciones concretas. Estos módulos son:

- **Interleaved dataframe**: Este módulo realiza varias transformaciones a los datos de entrada. La primera transformación consiste en separar el contenido relativo a orientaciones del relativo a posiciones dentro de los datos de entrada, es decir, separa los vectores de 3 y de 4 componentes. Después, realiza una operación similar a una trasposición matricial con el fin de exportar estos datos a Unity, lo cual no es de importancia en nuestro caso.
- **Image builder & Image enricher**: Las transformaciones del *image builder* tienen el fin de conformar las imágenes finales que se emplearan como inputs en nuestra red neuronal. Gracias a *image enricher* podemos obtener un mayo número de imágenes empleando diferentes algoritmos de *data augmentation*:
  - **Overlap**: Se basa en construir imágenes con muestras compartidas con la imagen formada anteriormente. Esto permite aumentar considerablemente el número de ejemplos para alimentar nuestra red, pero por otro lado aumenta la correlación entre los propios ejemplos, pudiendo propiciar que la red neuronal sobreentrene y termine memorizando los ejemplos.
  - **Rotaciones**: Esta técnica se basa en aplicar productos de cuaterniones a los datos preprocesados simulando que también han sido grabados empleando diferentes direcciones.
  - **FFT bidimensional**: Este algoritmo se basa en aplicar la Transformada de Fourier bidimensional a la imagen generada y concatenar el resultado de la operación, es decir parte real e imaginaria, a la imagen original. Esto sirve como un aporte de información a la red

Tras analizar la situación relativa a los preprocesamientos de los ficheros de la base de datos nos dimos cuenta de que, debido a la diferente duración de los ejercicios grabados, se va a producir un desbalanceo entre en número de ejemplos dentro de cada ejercicio. Este suceso es mucho más perjudicial en los ejercicios de tren inferior, ya que las actividades A01, A02 y A03 tienen una duración muy superior a A04. En los ejercicios de tren superior, este hecho es menos

notables, ya que los ejercicios suelen tener una duración más equiparable, a excepción de A12 y A13 que tienen una duración media un 30 % inferior al resto de los ejercicios. Este desbalanceo del número de ejemplos entre actividades puede propiciar un aumento del *overfitting* en los entrenamientos.

Entorno de entrenamiento

En este entorno se lleva a cabo el entrenamiento de la red, especificando todos los hiperparámetros que se pueden modificar de la misma manera que en el entorno de preprocesado. También permite generar automáticamente métricas del desempeño de la red, así como matrices de confusión y guardar los datos del modelo entrenado.

Algunos de los hiperparámetros configurables pueden ser: la red neuronal basada en Tensorflow que deseamos emplear, los diferentes ejercicios que se van a clasificar, especificar los sujetos de entrenamiento, validación y test, el tamaño de las imágenes de entrada, diferentes *callbacks*, el número de canales, el número de *steps* de entrenamiento, validación y test o las épocas.

Inicialmente disponíamos de un número limitado de muestras y de sujetos. Gracias a los métodos de *data augmentation* se ha solventado en cierta medida el problema de las muestras de ejemplo limitadas. Para solucionar el problema del número de sujetos limitado, existe el procedimiento *k-fold cross validation* o la validación cruzada k-fold, el cual está implementado en el framework que vamos a usar.

K-Fold Cross Validation es una forma de evaluar los modelos de aprendizaje automático dado un conjunto limitado de muestras, la cual es nuestra situación. Es un método muy empleado en estos casos debido a su simplicidad y estimaciones no optimistas. El procedimiento consiste en los siguientes pasos [50]:

1. Las muestras se separan en k grupos.

2. Dentro de cada grupo:

   a) Se seleccionan las muestras que forman el conjunto de test, y el conjunto de entrenamiento.

   b) Se entrena una red neuronal con el conjunto de entrenamiento recién formado y se evalúa el modelo con el conjunto de test.

   c) Se almacenan las métricas de desempeño y se descarta el modelo.

3. Finalmente, la precisión del modelo conjunto es la media aritmética de las precisiones de los entrenamientos individuales.

ID	EJERCICIO	TIPO	PLANO	# REPETICIONES
A05	MoverVasoDerecha	Unimanual	Oblicuo a 45°	5
A06	MoverVasoIzquierda	Unimanual	Oblicuo a 45°	5
A07	BeberVasoDerecha	Unimanual	Oblicuo a 45°	5
A08	BeberVasoIzquierda	Unimanual	Oblicuo a 45°	5
A09	MontarLEGO	Bimanual	Oblicuo a 45°	1
A10	BalonAlAire	Bimanual	Oblicuo a 45°	10
A11	CogerBotellaAltaDerecha	Unimanual	Oblicuo a 45°	5
A12	CogerBotellaAltaIzquierda	Unimanual	Oblicuo a 45°	5
A13	RomperPapelBola	Bimanual	Oblicuo a 45°	1

Tabla 7: Ejercicios para clasificación de tren superior

5.2. HAR en actividades de tren superior

En este apartado del capítulo se exponen las diferentes pruebas que se han llevado a cabo con el framework mencionado anteriormente de entrenamiento para las actividades que emplean el tren superior. Estas actividades son las 9 recogidas en la tabla 7. Por otra parte, sólo es posible emplear los sujetos que tienen al menos un ejercicio de tren superior, es por esto que se han utilizado 15 de ellos: S01, S02, S05, S29, S30, S31, S32, S33, S34, S35, S36, S37, S38, S39 y S40.

Durante los entrenamientos llevados a cabo, tanto para tren superior como inferior, se han aplicado 2 *callbacks*. Los **callbacks** son objetos que realizan diferentes acciones en las distintas fases de un entrenamiento de redes neuronales. Nosotros hemos empleado los siguientes:

- **ModelCheckpoint**: Este *callback* permite almacenar el modelo en un punto del entrenamiento concreto. Nosotros lo configuramos para que guardara los pesos del modelo cuando éste alcanzara su mayor precisión en el entrenamiento.
- **EarlyStopping**: Este *callback* permite acelerar el tiempo de entrenamiento, haciendo que éste finalice cuando una métrica no mejore a lo largo de varias épocas. En nuestro caso, elegimos la precisión como métrica decisoria y le otorgamos un valor de 7 épocas a la paciencia.

Estos *callbacks* se han mantenido a lo largo de todas las pruebas, tanto en entrenamientos individuales como en los entrenamientos de validación cruzada con k-fold.

Se han llevado a cabo los siguientes entrenamientos individuales:

00. En el preprocesado se estableció el tamaño de ventana a 512 muestras y un *overlap* de 496 muestras, es decir, compartiendo un 95 % de las muestras correspondientes a la ventana

anterior. También, se empleó *data agumentation* añadiendo a las imágenes finales la FFT bidimensional. En lo relativo al entrenamiento, se eligieron entrenamientos de 90 épocas, *batches* de 90 imágenes y *steps* de 95 imágenes. La arquitectura de la red neuronal empleada es una red convolucional conformada por 4 capas convolucionales (convolución 2D + *Max Pooling*) de 64, 64, 128 y 256 mapas de características respectivamente y una capa densa.

01. Para este entrenamiento sólo se modificó la arquitectura de la red, añadiendo capas de normalización del *batch* en cada capa convolucional.

02. Para este entrenamiento sólo se modificó la arquitectura de la red, cambiando el número de mapas de características de las capas de la red anterior pasando de ser 64, 64, 128 y 256 a 64, 64, 64 y 128.

03. La siguiente prueba se hizo modificando la red empleada en 01, pero añadiendo capas de *dropout* de tasas 0.1, 0.1, 0.15 y 0.2 respectivamente.

04. Los cambios introducidos en esta prueba comienzan con los parámetros de entrenamiento, pasando a considerar *batches* de tamaño 128 imágenes en vez de 90 y *steps* de tamaño 62 en vez de 95. Esta decisión se llevó a cabo para que el número total de ejemplos disponibles sea el producto entre el tamaño del *batch* y el número de *steps*. Además, sobre la anterior arquitectura de la red, se eliminó la última capa, pasando a ser 3 capas convolucionales formadas con una convolución 2D, una capa de *max pooling*, una de *batch normalization* y una capa de *dropout*.

05. Para esta prueba se mantuvo la anterior arquitectura de la red. El cambio aplicado se produjo en el preprocesado, en concreto, en el tamaño del *overlap*, que pasó de 496 muestras (95 %) a 410 muestras (80 %). Esta decisión se tomó debido a que en todas las pruebas llevadas a cabo hasta este momento, la red era capaz de aprender los ejemplos y, por consiguiente, se producía *overfitting*. Con la modificación del *overlap* se consigue reducir la correlación entre las muestras y que le cueste un poco más aprenderlas. Seguramente, el problema del *overfitting* también se produzca porque no tuviéramos un número suficiente de ejemplos. Como consecuencia de la modiciación dle preprocesado, para mantener la relación entre *batch* y *steps* se modificaron también sus valores a 64 y 21 respectivamente.

06. El cambio aplicado en esta prueba afecta a la arquitectura de la red, a la que eliminó la última capa.

07. En este entrenamiento, se aumentó la tasa de las capas de *dropout*, que actualmente tenía un valor de 0.1 y pasó a ser de 0.2 para ambas.

08. En esta prueba se aumentó de nuevo la tasa de las capas de *dropout*, que pasó a ser de 0.5 para ambas capas de la red.

09. Para la siguiente prueba, se llevó a cabo otro algoritmo de *data augmentation* ya mencionado anteriormente, que es la rotación de los cuaterniones de las imágenes. Los ángulos empleados para rotar las imágenes fueron 0º, 15,º 30º, 45º, 60º, 75º, 90º, 105º, 120º, 135º, 165º y 180º. Debido a que el número de ejemplos aumentó, se tuvieron que modificar los

#	Entrenamiento		Validación		Test	
	Pérdida	Precisión	Pérdida	Precisión	Pérdida	Precisión
0	0.1228	0.9542	5.8737	0.428	4.9073	0.4039
1	0.0001	0.9998	4.0563	0.6685	2.5723	0.7097
2	0.0001	1	5.1876	0.7792	4.3526	0.772
3	0.0624	0.9836	4.3535	0.7763	5.6867	0.7639
4	0.002	0.9995	8.8789	0.6065	12.1525	0.6245
5	0.0579	0.9878	17.5594	0.7151	16.3545	0.6556
6	0.2611	0.9984	41.0579	0.8647	11.5851	0.8548
7	0.1367	0.9817	33.4847	0.7821	18.2697	0.7983
8	0.5917	0.9927	32.1857	0.7094	37.0151	0.7697
9	0.0001	0.9999	18.0396	0.83	17.2637	0.8203
10	0.0186	0.9944	5.9646	0.8095	14.7375	0.8384

Tabla 8: Resultados entrenamientos individuales tren superior

parámetros de entrenamiento, siendo ahora el tamaño de *batch* 128 muestras y el tamaño de los *steps* de 133 muestras.

10. El cambio que se aplicó en la última prueba fue la modificación de la tasa de las capas de *dropout*, que era de 0.5 y pasó a ser de 0.2.

Los resultados obtenidos en los entrenamientos individuales explicados anteriormente se recogen en la tabla 8, la cual muestra para cada entrenamiento individual las métricas de pérdida y precisión en las fases de entrenamiento validación y test.

Con respecto a las decisiones tomadas a lo largo de las pruebas para el problema de HAR en los ejercicios de tren superior podemos observar según las métricas expuestas en la tabla 8 lo siguiente: partiendo de la prueba llamada 00, la adición de la capa de normalización del *batch* supuso una mejora de más de un 30 % de precisión en los ejemplos de test. Otra mejora sustancial fue propiciada por la reducción del número de mapas de características propuesto en 02 y la reducción del tamaño de la red a únicamente 2 capas, aplicado en 06. En las pruebas mencionadas anteriormente, conseguimos un buen valor de la métrica de precisión alcanzando valores superiores al 80 %, pero con un valor de las pérdidas muy elevado, que ya en la prueba 10 se consigue reducir con un aumento en el número de ejemplos y retoques en las tasas de las capas de *dropout*.

A partir del último entrenamiento individual, llamado 10, llevado a cabo. Se procedió con un entrenamiento *K-fold cross validation*, explicado en el apartado 5.1, debido a que se consideró que era una red que respondía con buenas métricas para nuestra situación. En estos entrenamientos se prepararon todas las configuraciones posibles cumpliendo las siguientes condiciones:

- Sólo se tomará un sujeto para test y validación, que coincidirán.

- En cada grupo, sólo se emplearán como sujetos de test, aquellos que cuenten con todos los ejercicios de tren superior disponibles, quedándonos con S01, S02, S05, S29, S30, S31, S32, S33, S34, S35, S37, S39 y S40; es decir, 13 sujetos.

Se llevaron a cabo los siguientes entrenamientos de k-fold:

00. Para el primer entrenamiento con k-fold decidimos emplear la mitad de los entrenamientos individuales disponibles, es decir, 7 entrenamientos. Con esta situación, al ejecutar el entrenamiento en el *framework*, sucedió un error inesperado que imposibilitó la terminación del entrenamiento. Este error pudo deberse a la saturación de la memoria empleada por el *framework*.

01. Para el segundo entrenamiento empleando k-fold, queríamos evitar que sucediera de nuevo el error de la prueba anterior, por lo que limitamos el número de entrenamientos individuales llevando a cabo un 6-fold con la red empleada en el entrenamiento individual 10: una red de dos capas formadas por una convolución bidimensional de 64 mapas de características, una capa de *max pooling*, una capa de normalización de *batch* y una de *dropout* con tasa 0.2. Este entrenamiento consiguió una precisión del 98 % en las muestras de test. La matriz de confusión y las métricas agregadas de este entrenamiento con k-fold pueden observarse en las figuras 44 y 45.

Sobre la matriz de confusión normalizada de la figura 44 cabe destacar que el modelo que hemos obtenido es capaz de clasificar con una gran exactitud los ejercicios que se le presentan. Las actividades que más problemas tiene a la hora de la clasificación son la A06 o MoverVasoIzquierda, que lo confunde ligeramente con A10 o BalonAlAire; la A09 o MontarLEGO, que lo confunde ligeramente con A05 o MoverVasoDerecha; y la A13 o RomperPapelBola, que lo confunde ligeramente con A09 o MontarLEGO.

En lo respectivo a las métricas de la figura 45 vemos que son muy prometedoras con datos medios como sensitividad y especificidad de valores 89 % y 99 % respectivamente o un valor de precisión de la clasificación del 98 %.

A continuación, analizaremos los entrenamientos individuales del k-fold de tren superior que más problemas han tenido, expuestas en la figura 46:

- Para el primero de ellos: tras ver sus desempeños, hemos observado un entrenamiento que consigue un 76.18 % de precisión en los ejemplos de test. Atendiendo a su matriz de confusión normalizada vemos que los ejercicios con los que tiene más problemas identificando correctamente son A06 (MoverVasoIzquierda), que lo confunde completamente con A10 (BalonAlAire) y A13 (RomperPapelBola); A08(BeberVasoIzquierda), que lo confunde notablemente con A06; y A12(CogerBotellaAltaIzquierda), que lo confunde completamente con A08 y A13.

- Para el segundo de ellos, con un 76.94 % de precisión con los ejemplos de test: Vemos en su matriz de confusión normalizada que, los ejercicios con los que tiene más problemas

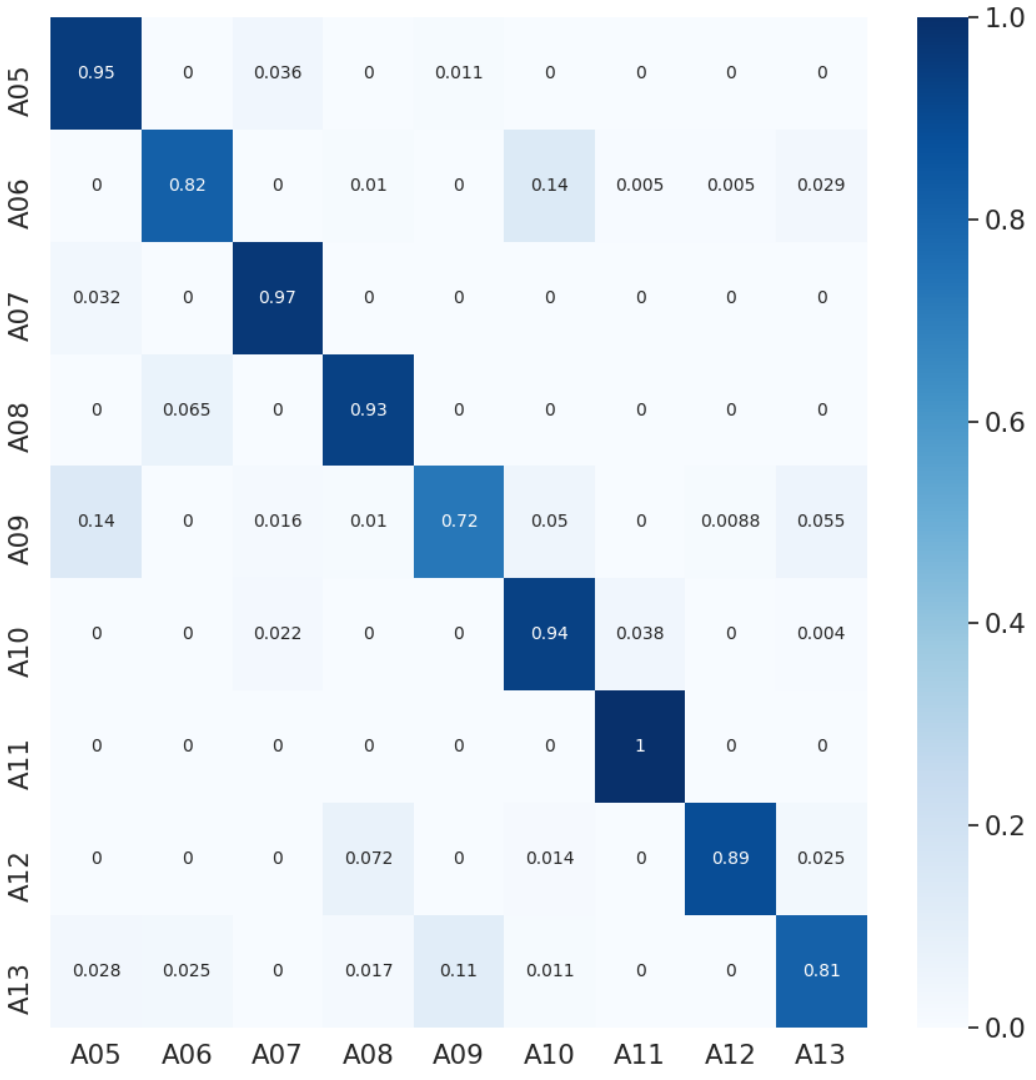

Figura 44: Matriz de confusión normalizada ejercicios tren superior

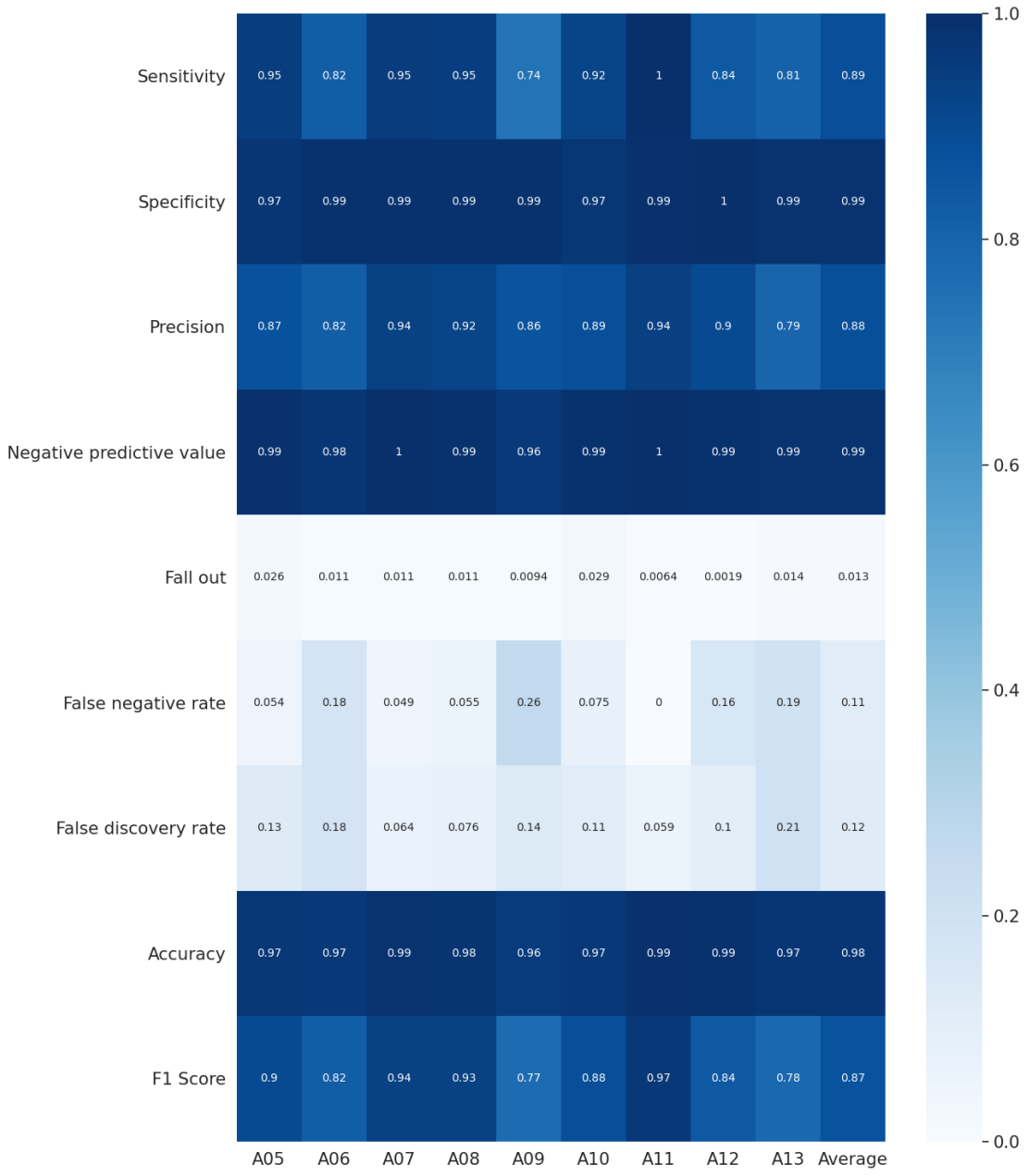

Figura 45: Matriz de métricas ejercicios tren superior

ID	EJERCICIO	TIPO	PLANO	# REPETICIONES
A01	AndarFrenteYVuelta	Pierna	Sagital	3
A02	AndarHaciaAtrasYVuelta	Pierna	Sagital	3
A03	AndarSobreLinea	Pierna	Oblicuo a 20°	3
A04	Sentadillas	Pierna	Oblicuo a 45°	5

Tabla 9: Ejercicios para clasificación de tren inferior

a la hora de la clasificación son A09 (MontarLEGO), que lo confunde mucho con A05 (MoverVasoDerecha); A10, que lo confunde con A11 (CogerBotellaAltaDerecha); y A13, que lo confunde notablemente con A05.

Vemos que los ejercicios con los que tiene mayor problema contienen algunos gestos similares a los ejercicios con los que los confunde, por lo que es algo aceptable. Aunque por otro lado, no ocurre con los mismos gestos hechos con ambas manos. Es decir, el ejercicio A09, contiene gestos similares a A05 y A06, pero no sucede que confunda simultáneamente A09 con A05 y A09 con A06.

5.3. HAR en actividades de tren inferior

En este apartado del capítulo se exponen las diferentes pruebas que se han llevado a cabo con el framework mencionado anteriormente de entrenamiento para las actividades que emplean el tren inferior. Estas actividades son las 4 recogidas en la tabla 9. Por otra parte, sólo es posible emplear los sujetos que tienen al menos un ejercicio de tren superior, es por esto que se han utilizado 14 de ellos: S01, S02, S03, S05, S29, S30, S31, S32, S33, S34, S35, S36, S37, S38.

Las pruebas que se han llevado a cabo para conseguir el reconocimiento de las actividades de tren inferior han sido las siguientes:

00. En la primera prueba de los entrenamientos individuales se mantuvo la configuración empleada en el preprocesado de los ejercicios de tren superior: imágenes de 512 muestras con un *overlap* de 410 muestras (80 %). De la misma manera, se mantienen las configuraciones de entrenamiento anteriores, siendo el tamaño del *batch* 128 imágenes, el tamaño de los *steps* 133 imágenes y el número de épocas 35. Para la arquitectura de la red, comenzamos probando la última configuración que se probó en los ejercicios de tren superior, ya que había dado buenos resultados. Esta red consiste en dos capas convolucionales formadas por: una capa de convolución bidimensional con 64 mapas de características, una capa de *max pooling*, una de normalización del *batch* y una capa de *dropout* de tasa 0.2. Tras las capas convolucionales hay una capa densa para obtener las salidas de la red.

01. En la primera prueba, la red sufría de *overfitting*, por lo que probamos reduciendo el número de capas a 1 y reduciendo el número de mapas de características a 32, manteniendo todo lo

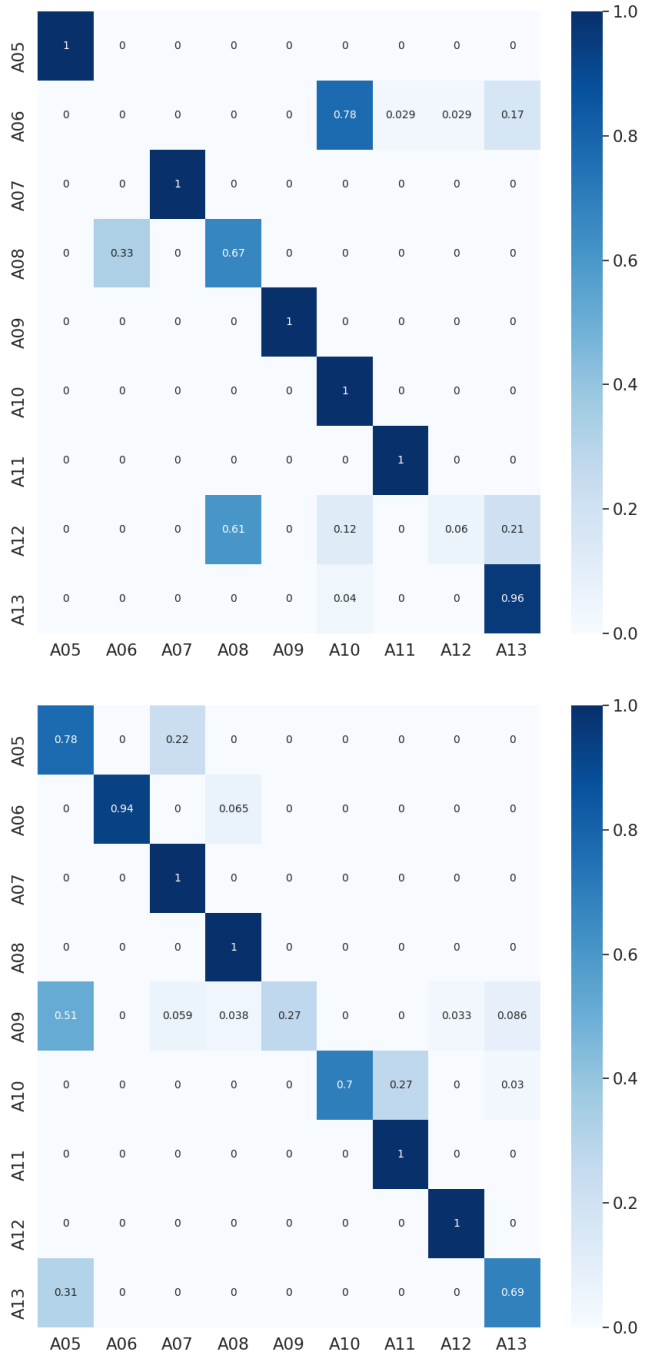

Figura 46: Matrices de confusión de los peores entrenamientos individuales del k-fold

#	Entrenamiento		Validación		Test	
	Pérdida	Precisión	Pérdida	Precisión	Pérdida	Precisión
0	0.0353	0.9917	14.876	0.5095	7.6556	0.5135
1	0.0277	0.9986	90.0371	0.4759	52.9993	0.4383
2	0.0298	0.9994	108.4739	0.4478	103.9126	0.4753
3	0.1751	0.983	9.5061	0.5618	9.2662	0.5749
4	0.0463	0.9944	13.0071	0.6025	8.0246	0.5996

Tabla 10: Resultados entrenamientos individuales tren inferior

demás tal y como estaba. Esta acción se llevó a cabo pese a que las redes neuronales con menos de 2 capas no se consideren profundas, pero queríamos comprobar los resultados de esta nueva red.

02. En la siguiente prueba de entrenamiento individual probamos a mantener la arquitectura anterior, pero aumentando la tasa de *dropout* de 0.2 a 0.25. Pese a estas decisiones, veíamos que se seguía produciendo *overfitting*, aunque en menos medida que en el primer caso.

03. Para la siguiente prueba, se tomó como base la red empleada en 01. El cambio que se llevó a cabo fue emplear canales. Los **canales** en las redes convolucionales suponen tratar las imágenes al igual que se tratan las imágenes en formato RGB. Esto supone que las entradas pasan a ser tensores, compuestos por las imágenes iniciales bidimensionales de cada componente de los cuaterniones $[w, x, y, z]$ "apiladas". También, esto supone que los mapas de características y los filtros empleados ganan una dimensión más de tamaño el número de canales que empleamos. En nuestro caso, como los cuaterniones cuentan con 4 componentes, emplearemos 4 canales. De esta manera, conseguimos que cada componente de los cuaterniones se trate de una forma más independiente del resto de componentes, lo cual puede beneficiar a la identificación de patrones de movimiento dentro de los ejercicios que deseamos clasificar.

04. Finalmente, en la última prueba individual, partiendo de la arquitectura usada en el entrenamiento anterior, se redujo el número de mapas de características de 32 a 16, manteniendo todas las demás configuraciones de preprocesado, entrenamiento y arquitectura de la red convolucional.

Los resultado obtenidos en los entrenamientos individuales que acaban de ser descritos se han recogido en la tabla 10, donde se muestras las métricas de pérdida y precisión de los entrenamientos individuales en los sets de entrenamiento, validación y test.

A partir de los resultados expuestos en la tabla 10 y de las decisiones tomadas en cada una de las pruebas, podemos ver que, partiendo de la red neuronal que dio los mejores resultados para la clasificación individuales en tren superior, sólo obtenemos un 51.35 % de precisión. Hemos pasado de tener 9 ejercicios a 4, pero también es verdad que A01 (AndarFrenteYVuelta) y A03 (AndarSobreLinea) contienen gesto muy similares, lo que puede hacer que éstos se confundan con facilidad. Además, Debido a el aumento de datos mediante las rotaciones de los

Figura 47: Matriz de confusión normalizada ejercicios tren inferior (k-fold 00)

cuaterniones, puede haber provocado que para clasificar A01, A02 (AndarHaciaAtrasYVuelta) y A03 exista todavía más dificultades. Volviendo a los resultados de los entrenamientos individuales, observamos que la reducción de las dimensiones de la red perjudican a los resultados, y el uso de los canales mejora las métricas.

Para finalizar la fase de las pruebas para el tren inferior, presentaremos las pruebas que se llevaron a cabo empleando k-fold:

00. Para el primer entrenamiento de k-fold, empleamos la red usada en el entrenamiento de k-fold 01 para tren superior de dos capas (convolución 2D, *mas pooling*, normalización del *batch* y *dropout* de tasa 0.2). En este caso, empleamos 8 sujetos de los 14 disponibles, por si sucedía el mismo problema que con el primer entrenamiento con k-fold. La precisión obtenida en este caso es de un 82 %, lo cual es muy inferior a lo obtenido en el tren superior, aunque no es un mal resultado. Tanto la matriz de confusión normalizada como las métricas resultantes pueden observarse en las figuras 47 y 48.

01. Para este segundo entrenamiento, consideramos que sería interesante comprobar si el framework admitía loas 14 entrenamientos individuales, ya que son muchos menos datos los que se tienen que cargar en la RAM al haber sólo 4 actividades. También decidimos emplear la misma red que en el k-fold anterior, pero añadiendo los 4 canales. La precisión obtenida en este entrenamiento ha descendido ligeramente al 0.79 %.

02. Finalmente, en el último entrenamiento decidimos probar con una red similar a las anterio-

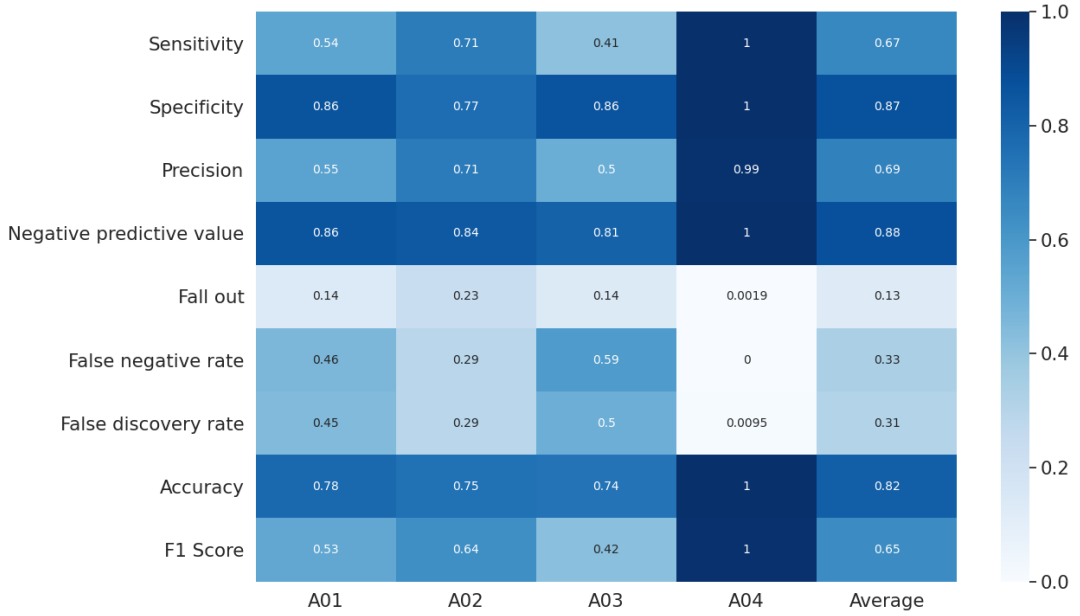

Figura 48: Matriz de métricas ejercicios tren inferior (k-fold 00)

Figura 49: Matriz de confusión normalizada ejercicios tren inferior (k-fold 01)

Figura 50: Matriz de métricas ejercicios tren inferior (k-fold 01)

res, pero con una única capa y reduciendo el número de mapas de características de 64 a 16. También, se utilizaron 7 sujetos. Los resultados de esta prueba superaron mínimamente a los dos anteriores, consiguiendo una precisión media de 83 %. De nuevo, al tratarse de una red con una única capa, no sería considerada profunda, por lo que los resultados no son realmente importantes para este proyecto.

A continuación analizaremos algunos de los entrenamientos individuales del k-fold 01. En concreto, analizaremos los 4 con peores medidas de precisión:

- El primer entrenamiento individual que analizaremos emplea como sujeto de test al sujeto S30. En la matriz de confusión de la izquierda de la figura 51, observamos que la red tiene serios problemas identificando las actividades de caminar (A01, A02 y A03). Podemos observar que las actividades A02 y A03 las confunde entre sí, y no es capaz de identificar con más de un 20 % de precisión la actividad A01. Llama la atención que para A02 se confunda con un 23 % de error la actividad A04, ya que en casi ninguno de los entrenamientos individuales las actividades A01, A02 y A03 se han confundido con ella.

- Para el segundo entrenamiento individual, empleamos al sujeto S33 como sujeto de test. En este caso, y según la matriz de confusión derecha en la figura 51 observamos que la red tiende a confundir las actividades A01 y A03, y no es capaz de identificar de una manera clara la actividad A02, confundiéndola con los otros dos ejercicios que implican caminar.

- En el siguiente entrenamiento, se emplea como sujeto de test al sujeto S29. Observando la matriz izquierda de la figura 52. En este caso, nos llama la atención que clasifique correctamente la actividad A01 con tanta tasa de aciertos, aunque un 38 % de confusión

Figura 51: Matrices de confusión normalizadas entrenamientos individuales 1 (izda) y 2 (dcha)

Figura 52: Matrices de confusión normalizadas entrenamientos individuales 3 (izda) y 4 (dcha)

con la actividad A03 no es para nada despreciable. También vemos que el ejercicio A03 es clasificado erróneamente con el A01 con una tasa de fallo muy elevada. Por último, la actividad A02 no es capaz de tener una identificación fiable, ya que la confunde con la misma probabilidad que A01 y A02, y con el doble de probabilidad con A03.

- Finalmente, el último entrenamiento individual del k-fold que analizaremos toma como sujeto de test al sujeto S31. Podemos ver que hasta en los entrenamientos individuales con peores métricas, la red no tiene ningún problema con la identificación de la actividad A04, ya que es suficientemente diferente a los otros 3 ejercicios como para que estas confusiones no se cometan. En lo correspondiente a este entrenamiento, según vemos en la matriz derecha de la figura 52, la actividad A02 tiene una tasa de acierto superior a cualquiera de los entrenamientos analizados, aunque no deja de cometer casi la mitad de errores al clasificarla. Podemos ver que la actividad A03 es confundida con el ejercicio A02, teniendo métricas de acierto similares. En cuanto a la actividad A01, observamos que lo confunde con el ejercicio A03.

Capítulo 6

Conclusiones, Presupuesto y Líneas Futuras

6.1. Conclusiones

En este proyecto se pretendía emplear sensores inerciales para grabar ejercicios realizables en la vida cotidiana en entornos no clínicos, conformar una base de datos tras procesar las grabaciones y comprobar si con ellos podíamos llevar a cabo el reconocimiento de esas actividades mediante las primeras pruebas empleando redes neuronales de aprendizaje profundo.

De los objetivos marcados en el apartado 1.2: Se ha conseguido realizar grabaciones y llevar a cabo la extracción de métricas angulares útiles a partir de los sensores inerciales empleados. También se ha formalizado una base de datos con esos movimientos tras el procesado de las grabaciones. Esta base de datos ha sido creada con el fin de ser publicada y que diferentes estudiantes, investigadores o cualquier usuario pueda acceder a ella y utilizarla de la forma más sencilla posible. Finalmente, se ha conseguido llevar a cabo el reconocimiento de las actividades grabadas empleando una CNN o red neuronal convolucional, utilizando para ello un framework especializado en clasificación orientado a HAR. Se han utilizado técnicas de *data augmentation* como el aplicado de un *overlap*, la rotación de los cuaterniones grabados y la adición de la FFT bidimensional a los *inputs* de la red. Para evitar el *overfitting* se emplearon algunas de las técnicas mencionadas teóricamente en el apartado 3.3.4, tales como la disminución del tamaño de la red, *dropout* o normalización del *batch*. El problema de clasificación se separó en ejercicios de tren superior, donde el modelo entrenado es capaz de clasificar 9 actividades diferentes con una precisión del 98 %, mientras que en los ejercicios de tren inferior, el modelo entrenado es capaz de clasificar 4 actividades correctamente con una precisión entorno al 80 %.

Tras la realización de todo el estudio, investigaciones y las pruebas que envuelven a este proyecto, hemos llegado a diferentes conclusiones:

- Actualmente, los problemas de HAR tienen soluciones muy sencillas gracias a todos

NOMBRE	PRECIO UNITARIO [€]	UNIDADES	SUBTOTAL [€]
Ordenador de uso amortizado	828.33	1	828.33
Sensores XSens DOT	180	5	900
Sensores TwynSens	-	-	-
Horas de trabajo	11.25	500	5625
Total			7353.33

Tabla 11: Presupuesto para el proyecto

los estudios y técnicas, tanto de *Machine Learning* clásico como de *Deep Learning*, desarrolladas en los últimos años. Estas técnicas han conseguido muy buenos resultados en problemas muy diferentes y de muy diferentes dificultades. Los problemas que se están tratando de sobrepasar es la utilización de modelos igual de potentes mediante sistemas que requieran de muchos menos recursos, que sean más cómodos de usar o que empleen modelos más simples.

- Como se ha mencionado previamente, hemos conseguido formar una base de datos con actividades cotidianas con vista a publicarla. Esta base de datos servirá a cualquier usuario que lo requiera para probar diferentes sistemas de HAR.

- Se ha hecho un primer intento de clasificación empleando una red neuronal convolucional. Las arquitecturas de redes neuronales para HAR empleadas no son las más potentes, precisas o las más novedosas, pero gracias a ellas hemos demostrado que la base de datos grabada tiene utilidad para probar sistemas de clasificación empleando redes neuronales.

- Los resultados de estas pruebas de clasificación han dado resultados positivos, con tasas de precisión superiores al 80 %.

6.2. Presupuesto

En este apartado del capítulo final de TFG se mostrarán diferentes activos empleados para el desarrollo del proyecto así como cuánto ha sido la inversión para llevarlo a cabo.

Como vemos en la tabla 11, existen diferentes activos, que han sido mencionados previamente en el apartado 1.4. El ordenador es el empleado para los entrenamientos de las redes neuronales que se encuentra en la ETS de Ingenieros de Telecomunicación, con un precio total de 2485€ a amortizar en 3 años. Los sensores empleados para unas pruebas previas a la grabación de la base de datos fueron los XSens DOT. Estos sensores tienen un precio de 5 unidades a 900€. Los sensores empleados para la grabación de la base de datos han sido los TwynSens que, al ser un producto de diferentes proyectos del grupo de investigación GTI, no podemos saber cuál es su valor monetario. Finalmente, en lo relativo a las horas de trabajo, hacemos una estimación sobre todo el tiempo que ha sido necesario para realizar la totalidad del proyecto y hemos llegado a

la conclusión de que han sido unas 500 horas aproximadamente. Según el Boletín Oficial del Estado, para el año 2021, los sueldos base para ingenieros son de 1.799,68€ mensuales, lo que, con una jornada laboral de 40 horas mensuales, hace un total de 11,25€ a la hora.

6.3. Líneas Futuras

Para finalizar, queremos dejar constancia en este apartado de las diferentes posibilidades de continuación y mejora tras el desarrollo de este proyecto.

Inicialmente, lo más inmediato podría ser intentar mejorar las métricas de los entrenamientos haciendo modificaciones de las redes que hemos usado en el proyecto, dado que hemos contado con un tiempo reducido para hacerlas y no hemos podido hacer todas las que nos hubiera gustado. También podría ser interesante ver si podríamos hacer entrenamientos de k-fold con todos los conjuntos posibles, analizar cada uno de los entrenamientos de los k-fold o incluso intentar conseguir un único modelos para clasificar tanto actividades de tren superior como inferior.

En posteriores pruebas, podría también utilizarse el framework empleando como inputs vectores tridimensionales, los cuales contendrían los ángulos de Euler extraídos a partir de los ficheros de la base de datos empleando las técnicas explicadas en 3.2.4 y 3.2.5.

Una de las líneas de trabajo pudiera ser el empleo de otro tipo de redes neuronales, como las recurrentes e incluso emplear celdas LSTM o GRU, dado que las empleadas en las primeras pruebas (CNN) no suponen ninguna novedad dentro de todos los tipos de arquitecturas de redes neuronales utilizadas actualmente.

Otra línea podría ser la utilización de redes GAN (*Generative Adversary Network*) para aumentar los ejemplos con los que alimentar a las redes de datos para ver si podemos mejorar el desempeño de las redes, sobre todo en la clasificación de ejercicios de tren inferior.

Finalmente, podríamos plantear ampliar la base de datos con un mayor número de sujetos, dado que con 14, no son un gran número. Además, podría buscarse una solución al problema de desbalanceo controlando el tiempo de duración de los ejercicios, aunque podría haber diferentes soluciones.

Glosario

AI *Artificial Intelligence* | Inteligencia Artificial. 30, 31, 33

AR *Agumented Reality* | Realidad Aumentada. 48

CNN *Convolutional Neural Network* | Red Neuronal Convolucional. 12, 13, 78, 80

CSS *Cascading Stylesheets* | Hoja de Estilos en Cascada. 7

CWT *Continuous Wavelet Transform* | Transformada de Wavelet Continua. 4

DL *Deep Learning* | Aprendizaje Profundo. 1, 7, 13, 30, 34, 35, 37

DRNN *Deep Recurrent Neural Network* | Red Neuronal Recurrente Profunda. 12

DT *Decision Tree* | Árbol de Decisión. 11, 12, 33

EMG *Electromyogram* | Electromiograma. 4

ETS - | Escuela Técnica Superior. 15, 79

ETSIT - | Escuela Técnica Superior de Ingenieros de Telecomunicación. 1, 8

FFT *Fast Fourier Transform* | Transformada Rápida de Fourier. 3, 63, 66, 78

GPU *Graphic Processing Unit* | Unidad de Procesamiento Gráfico. 2

GRU *Gated Recurrent Unit* | Unidad Recurrente con Puertas. 41, 45, 80

GTI - | Grupo de Telemática e Imagen. 1, 5, 8, 15, 79

HAR *Human Activity Recognition* | Reconocimiento de Actividades Humanas. 2–4, 7, 9–12, 14, 42, 47, 48, 62, 67, 78, 79

HTML *Hypertext Mark Language* | Lenguaje de Marcas en Hipertexto. 7

IMU *Inertial Measurement Unit* | Unidad de Medida Inercial. 1, 5–8, 10–12, 14, 15, 42, 47, 51–54, 56–59

Glosario

IoT *Internet of the Things* | Internet de las Cosas. 2

KNN *K-Nearest Neighbors* | K Vecinos más Próximos. 11, 34

LLA *Left Lower Arm* | Antebrazo Izquierdo. 19

LLL *Left Lower Leg* | Pantorrilla Izquierda. 19, 29

LR *Logistic Regression* | Regresión Logística. 13

LSTM *Long Short-Term Memory* | Memoria Grande a Corto Plazo. 12, 13, 39, 41, 44, 45, 80

LUA *Left Upper Arm* | Brazo Izquierdo. 19

LUL *Left Upper Leg* | Muslo Izquierdo. 19, 29

ML *Machine Learning* | Aprendizaje Automático. 1, 12, 13, 30, 32, 33, 39

MLP *Multi-Layer Perceptron* | Perceptrón Multicapa. 34

PSD *Power Spectral Density* | Densidad Espectral de Potencia. 12

ReLU *Rectified Linear Unit* | Unidad Lineal Rectificada. 37, 39

RFo *Random Forest* | Bosque de Decisión Aleatorio. 12, 33

RGBD *Red-Green-Blue-Depth* | Rojo-Verde-Azul-Profundidad. 4

RLA *Right Lower Arm* | Antebrazo Derecho. 19

RLL *Right Lower Leg* | Pantorrilla Derecha. 19

RMSE *Root Mean Square Error* | Raíz del Error Cuadrático Medio. 50, 52, 54–59, 61

RNN *Recurrent Neural Network* | Red Neuronal Recurrente. 12, 13, 44, 45

RUA *Right Upper Arm* | Brazo Derecho. 19

RUL *Right Upper Leg* | Muslo Derecho. 19

SDK *Software Development Kit* | Kit de Desarrollo de Software. 48

STFT *Short-Time Fourier Transform* | Transformada Rápida de Fourier. 4, 12

SVM *Support Vector Machine* | Máquina de Vector de Soporte. 11, 12, 34

TFG - | Trabajo de Fin de Grado. 7, 8, 10, 31, 33, 34, 79

TPU *Tensorial Processing Unit* | Unidad de Procesamiento Tensorial. 2

UVa - | Universidad de Valladolid. 1, 8

Bibliografía

[1] X. Chen and T. Han, "Disruptive Technology Forecasting based on Gartner Hype Cycle," *2019 IEEE Technology & Engineering Management Conference (TEMSCON)*, 2019.

[2] Z. Meng, M. Zhang, C. Guo, Q. Fan, H. Zhang, N. Gao, and Z. Zhang, "Recent Progress in Sensing and Computing Techniques for Human Activity Recognition and Motion Analysis," *Electronics*, vol. 9, no. 9, p. 1357, 2020.

[3] A. Ng, "Machine Learning on Coursera," 2011.

[4] J. González Alonso and M. Martínez Zarzuela, " Periféricos basados en Arduino para interacción con sistemas médicos de simulación y rehabilitación," tech. rep., Universidad de Valladolid, 2017.

[5] J. González Alonso and M. Martínez Zarzuela, "Diseño y evaluación de un sistema vestible para captura de movimientos orientado a aplicaciones de evaluación ergonómica y monitorización de terapias de rehabilitación física," tech. rep., Universidad de Valladolid, 09 2018.

[6] "Quaternions Visualisation," 2021.

[7] F. Chollet, *Deep Learning with Python*. Manning Publications, 2017.

[8] E. Matthes, *Python Crash Course*. Amsterdam, Países Bajos: Amsterdam University Press, 2019.

[9] Xsens Technologies B.V.", "Xsens DOT User Manual," Tech. Rep. XD0502P, Xsens Technologies B.V.", 12 2020.

[10] "Xsens DOT Developer Conference: Automated Manual Tasks Risk Assessment, a benefit or a distraction?," 2021.

[11] D. Pérez de la Fuente and M. Martínez Zarzuela, " Análisis y clasificación de movimientos en actividades humanas a partir de vídeo 2D: adquisición de base de datos propia y comparativa de técnicas con aprendizaje profundo," tech. rep., Universidad de Valladolid, 2021.

[12] G. Pardo and M. Martínez Zarzuela, "Redes de Aprendizaje Profundo para reconocimiento de actividades humanas: framework de pre-procesado y entrenamiento en TensorFlow," tech. rep., Universidad de Valladolid", 2021.

[13] S. Jung, M. Michaud, L. Oudre, E. Dorveaux, L. Gorintin, N. Vayatis, and D. Ricard, "The Use of Inertial Measurement Units for the Study of Free Living Environment Activity Assessment: A Literature Review," *Sensors*, vol. 20, no. 19, p. 5625, 2020.

[14] D. Au, A. G. Matthew, P. Lopez, W. J. Hilton, R. Awasthi, G. Bousquet-Dion, K. Ladha, F. Carli, and D. Santa Mina, "Prehabilitation and acute postoperative physical activity in patients undergoing radical prostatectomy: a secondary analysis from an RCT," *Sports Medicine - Open*, vol. 5, no. 1, 2019.

[15] L. Carcreff, C. N. Gerber, A. Paraschiv-Ionescu, G. De Coulon, C. J. Newman, K. Aminian, and S. Armand, "Comparison of gait characteristics between clinical and daily life settings in children with cerebral palsy," *Scientific Reports*, vol. 10, no. 1, 2020.

[16] K. ELLIS, J. KERR, S. GODBOLE, J. STAUDENMAYER, and G. LANCKRIET, "Hip and Wrist Accelerometer Algorithms for Free-Living Behavior Classification," *Medicine & Science in Sports & Exercise*, vol. 48, no. 5, pp. 933–940, 2016.

[17] L. Fiorini, M. Bonaccorsi, S. Betti, D. Esposito, and F. Cavallo, "Combining wearable physiological and inertial sensors with indoor user localization network to enhance activity recognition," *Journal of Ambient Intelligence and Smart Environments*, vol. 10, no. 4, pp. 345–357, 2018.

[18] Y. J. Kim, K. D. Kim, S. H. Kim, S. Lee, and H. S. Lee, "Golf swing analysis system with a dual band and motion analysis algorithm," *IEEE Transactions on Consumer Electronics*, vol. 63, no. 3, pp. 309–317, 2017.

[19] F. Dadashi, G. Millet, and K. Aminian, "Gaussian process framework for pervasive estimation of swimming velocity with body-worn IMU," *Electronics Letters*, vol. 49, no. 1, pp. 44–45, 2013.

[20] Y. Wang, M. Chen, X. Wang, R. H. M. Chan, and W. J. Li, "IoT for Next-Generation Racket Sports Training," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4558–4566, 2018.

[21] Z. Fu, X. He, E. Wang, J. Huo, J. Huang, and D. Wu, "Personalized Human Activity Recognition Based on Integrated Wearable Sensor and Transfer Learning," *Sensors*, vol. 21, no. 3, p. 885, 2021.

[22] A. Ayman, O. Attalah, and H. Shaban, "An Efficient Human Activity Recognition Framework Based on Wearable IMU Wrist Sensors," *2019 IEEE International Conference on Imaging Systems and Techniques (IST)*, 2019.

[23] X. Li, Y. Wang, B. Zhang, and J. Ma, "PSDRNN: An Efficient and Effective HAR Scheme Based on Feature Extraction and Deep Learning," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6703–6713, 2020.

[24] A. Gumaei, M. Al-Rakhami, H. AlSalman, S. Md. Mizanur Rahman, and A. Alamri, "DL-HAR: Deep Learning-Based Human Activity Recognition Framework for Edge Computing," *Computers, Materials & Continua*, vol. 65, no. 2, pp. 1033–1057, 2020.

[25] A. Syed, Z. Sherhan, and A. Khalil, "Continuous Human Activity Recognition in Logistics from Inertial Sensor Data using Temporal Convolutions in CNN," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 10, 2020.

[26] S. Wan, L. Qi, X. Xu, C. Tong, and Z. Gu, "Deep Learning Models for Real-time Human Activity Recognition with Smartphones," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 743–755, 2019.

[27] C. Phyo, T. Zin, and P. Tin, "Deep Learning for Recognizing Human Activities Using Motions of Skeletal Joints," *IEEE Transactions on Consumer Electronics*, vol. 65, no. 2, pp. 243–252, 2019.

[28] P. Agarwal and M. Alam, "A Lightweight Deep Learning Model for Human Activity Recognition on Edge Devices," *Procedia Computer Science*, vol. 167, pp. 2364–2373, 2020.

[29] S. Sáez Bombín and M. Martínez Zarzuela, " Reconocimiento de actividades físicas con sensores inerciales y Redes Neuronales de Aprendizaje Profundo," tech. rep., Universidad de Valladolid, 2018.

[30] S. Sáez Bombín and M. Martínez Zarzuela, " Sistema de Aprendizaje Profundo para reconocimiento de actividades con sensores de captura de movimientos," tech. rep., Universidad de Valladolid, 2020.

[31] L. Sy, "Replication Data for Estimating Lower Limb Kinematics using a Reduced Wearable Sensor Count," 2019.

[32] O. Banos, A. Toth, and O. Amft, "REALDISP Activity Recognition Dataset Data Set," 2014.

[33] P. Kelland and P. Tait, *Introduction to Quaternions, by P. Kelland and P.G. Tait*. University of Edinburgh, 1881.

[34] D. Rowenhorst, A. D. Rollett, G. S. Rohrer, M. Groeber, M. Jackson, P. J. Konijnenberg, and M. De Graef, "Consistent representations of and conversions between 3D rotations," *Modelling and Simulation in Materials Science and Engineering*, vol. 23, no. 8, p. 083501, 2015.

[35] A. C. Robinson, "On the Use of Quaternions in Simulation of a Rigid-Body Motion," tech. rep., Aeronautical Research Laboratory, 12 1958.

[36] J. Winans, "Quaternion Physical Quantities," *Foundations of Physics*, vol. 7, no. 5-6, pp. 341–349, 1977.

[37] S. L. Altmann, *Rotations, Quaternions, and Double Groups*. Dover Publications, 2013.

[38] T. Jespersen, "quat2eul.m MATLAB Function Implementation," 2020.

[39] M. Tincknell, "Class Quaternion on MatLab," 2017.

[40] D. Eberly, "Euler Angle Formulas," tech. rep., Geometric Tools, 12 1999.

[41] H. Kurokawa, “A Geometric Study of Single Gimbal Control Moment Gyros,” Tech. Rep. 175, Agency of Industrial Technology and Science, Ministry of International Trade and Industry, Japan, 6 1997.

[42] K. Shoemake, “Animating rotation with quaternion curves,” *Proceedings of the 12th annual conference on Computer graphics and interactive techniques - SIGGRAPH ’85*, 1985.

[43] K. Shoemake, “Quaternions,” *Department of Computer and Information Science. University of Pennsylvania*, 1992.

[44] Real Academia Española", “Diccionario de la lengua española,” 1780.

[45] S. Russell and P. Norvig, *Artificial Intelligence*. Pearson, 2016.

[46] Stanford University.ªnd JMcCarthy, “WHAT IS ARTIFICIAL INTELLIGENCE?,” tech. rep., Stanford University, 11 2004.

[47] D. Dobrev, “A Definition of Artificial Intelligence,” tech. rep., Institute of Mathematics and Informatics: Sofia, BG, 2005.

[48] O. Simeone, “A Brief Introduction to Machine Learning for Engineers,” *Foundations and Trends® in Signal Processing*, vol. 12, no. 3-4, pp. 200–431, 2018.

[49] B. Ding, H. Qian, and J. Zhou, “Activation functions and their characteristics in deep neural networks,” *2018 Chinese Control And Decision Conference (CCDC)*, 2018.

[50] S. R. S. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[51] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *The Journal of Machine Learning Research*, 2014.

[52] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, pp. 448–456, 07–09 Jul 2015.

[53] NVIDIA", “NVIDIA Maxine Documentation,” 2020.