UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Trabajo Fin de Máster del
Máster de Ingeniería y Ciencia de Datos

Desarrollo de un modelo de reconocimiento de acciones en procesos industriales

Javier de la Fuente Casal

Dirigido por: Olga Santos Martín

Curso: 2022-2023: Convocatoria septiembre

Agradecimientos

Quisiera agradecer a toda la familia y amigos que me han apoyado a lo largo de este máster. A todos aquellos profesores que me han descubierto este mundo con tanta profundidad y me han dado la motivación para seguir aprendiendo.

Abstract

Human action recognition has been a topic of great interest in recent decades due to its wide range of applications. Among them, its application in industrial environments, which had been more limited until now due to data collection difficulties, offers significant opportunities in terms of increased traceability, safety control, protocol adherence, and operator performance. The OpenPack dataset provides a wealth of data for classifying actions in industrial settings, collected from multiple sensors in a controlled environment, and proposes a competition to develop a system capable of recognizing the actions performed by an operator.

This work aimed to employ a series of multi-modal combined models that process data from inertial sensors and visible cameras to classify each action performed by the operator, achieving results comparable to those of the top competitors who participated in the competition proposed by the dataset creators. To achieve this, we leveraged the features of convolutional neural networks and recurrent neural networks for data from inertial sensors, and graph-based neural networks for the visual stage through keypoint-based pose estimation.

Palabras clave: human action recognition, dataset, deep learning, machine learning, convolutional neural networks, recurrent neural networks, graph convolutional networks, keypoints, pose estimation, inertial sensors, LiDAR, depth image

Resumen

El reconocimiento de acciones humanas ha sido una problemática de gran interés en las últimas décadas debido a su gran cantidad de aplicaciones. Entre ellas, la aplicación a entornos industriales, que hasta ahora había sido más limitada debido a las dificultades de recolección de datos, brinda grandes oportunidades en cuanto a un aumento de la trazabilidad, control de seguridad, de protocolo y rendimiento del operario. El conjunto de datos OpenPack ofrece gran cantidad de datos para clasificar acciones en ambientes industriales recogidos a partir de múltiples sensores en un entorno controlado y propone un concurso para desarrollar un sistema capaz de reconocer la acción realizada por un operario.

En este trabajo se ha pretendido utilizar una serie de modelos combinados multi-modales que procesan los datos de sensores inerciales y cámaras visibles para clasificar cada acción realizada por el operario alcanzando resultados equiparables a los del top de competidores que han participado en el concurso propuesto por los creadores del conjunto de datos. Para ello se han aprovechado las características de las redes neuronales convolucionales y redes neuronales recurrentes para datos provenientes de sensores incerciales y redes neuronales basadas en grafos para la etapa visual a través de estimación de pose con *keypoints* dados.

Palabras clave: reconocimiento de acciones humanas, dataset, deep learning, aprendizaje automático, redes neuronales convolucionales, redes neuronales recurrentes, redes neuronales convolucionales basadas en grafos, keypoints, estimación de pose, sensores inerciales, LiDAR, imagen en profundidad

Índice general

Nomenclatura

BVP Blood volume pulse (Pulso de volumen sanguíneo), página 10

CNN Convolutional Neural Network, página 22

DPS Degrees per second (Grados por segundo), página 13

EDA Electrodermal activity (Actividad electro-dérmica), página 10

GCN Graph Convolutional Network, página 30

HAR Human Action Recognition (Reconocimiento de acciones humanas), página 1

HMM Hidden Markov Models, página 22

IMU Intertial Measurement Unit (Unidad de medición inercial), página 2

LiDAR Laser Imaging Detection and Ranging (Detector y medidor de imagen láser), página 2

LSTM Long Short Term Memory, página 32

RNN Recurrent Neural Networks, página 22

SOM Selft-Organizing Map, página 21

STIP Space-Time Interest Points, página 22

SVM Support Vector Machine, página 21

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación y objetivos

El reconocimiento de acciones en humanos (HAR) es un campo central en el área de la visión artificial y que puede ser aplicada en un amplio espectro de sectores como en salud, rehabilitación, entretenimiento, deportes o comprensión de vídeo (Vrigkas et al., 2015). Con el auge del IoT, se cuenta con una amplia gama de sensores, especialmente en *wearables* como pulseras de actividad y relojes inteligentes. A partir de los datos proporcionados por estos dispositivos se han creado numerosos sistemas que permiten reconocer actividades (Pham et al., 2020; Kalabakov et al., 2021). No solo con datos de sensores corporales se puede determinar una acción sino también a partir de imágenes y vídeos existen una serie de soluciones que se han ido explorando en los últimos años a partir de redes convolucionales de 3 dimensiones (Carreira and Zisserman, 2017; Tran et al., 2015) o añadiendo segmentaciones de temporalidad(Wang et al., 2016). Pero fue especialmente a partir de la extracción del esqueleto a partir de imagen (Toshev and Szegedy, 2014; Wei et al., 2016) cuando el reconocimiento mediante imagen comenzó a mejorar notablemente (Liu and Yuan, 2018; Zhang et al., 2022).

En entornos industriales como fábricas y centros logísticos, los trabajadores humanos siguen desempeñando roles importantes para garantizar respuestas flexibles a las demandas cambiantes de clientes y proveedores. La digitalización se está implementando ampliamente en estos entornos como parte de la transformación de la Industria 4.0, utilizando datos digitales para el reconocimiento de actividades humanas y para tomar decisiones comerciales basadas en información detallada del trabajo. El caso específico de reconocimiento de acciones en entornos industriales en el último lustro ha atraído la atención como tema de investigación activa (Qingxin et al., 2019; Xia et al., 2020b; Morales et al., 2022; Yoshimura et al., 2022a).

A pesar de todo, el estudio y desarrollo del reconocimiento de actividades en entornos industriales plantea varios desafíos significativos que deben ser abordados para obtener resultados precisos y aplicables en la práctica. A continuación, se detallan algunos de estos retos:

- Escasez de conjuntos de datos para dominios industriales: Uno de los principales desafíos es la falta de conjuntos de datos públicos adecuados para entrenar y evaluar algoritmos de reconocimiento de actividades en entornos industriales específicos. A menudo, los conjuntos de datos disponibles se centran en actividades cotidianas simples, como caminar y correr, y no reflejan las complejidades de las tareas realizadas en la industria, como tareas de fabricación y empaque. La escasez de conjuntos de datos con actividades industriales complejas limita la capacidad de desarrollar modelos precisos y generalizables para entornos de trabajo reales.

- Limitación de modalidades en los conjuntos de datos: Muchos conjuntos de datos públicos para tareas manuales proporcionan solo modalidades relacionadas con la visión, como imágenes o vídeos. Sin embargo, en entornos industriales, existen diferentes tipos de equipos de fabricación y sistemas de almacenamiento que pueden obstruir la visión y dificultar el uso de enfoques basados únicamente en la visión. La falta de datos multimodales que incluyan información de sensores inerciales (IMUs), imágenes de profundidad y nubes de puntos LiDAR, entre otros, dificulta la creación de modelos robustos y precisos para el reconocimiento de actividades industriales.

- Dificultades en la coordinación de datos humano-IoT: A medida que la digitalización avanza en los entornos industriales como parte de la Industria 4.0, se está utilizando una amplia gama de dispositivos habilitados para IoT que capturan datos sobre el movimiento humano y las operaciones realizadas. Sin embargo, coordinar y fusionar estos datos con los datos de sensores humanos puede ser un desafío debido a la variabilidad en el tiempo de registro y la escasez de lecturas de dispositivos IoT. Integrar de manera efectiva estas diferentes fuentes de datos para lograr un reconocimiento preciso de las actividades laborales es un desafío clave.

- Falta de metadatos detallados: Los metadatos asociados a los conjuntos de datos son información adicional que describe las características de las actividades y los sujetos involucrados. En el contexto industrial, los metadatos pueden incluir detalles sobre el nivel de experiencia de los trabajadores en tareas de embalaje, características físicas relevantes y otros detalles contextuales. La falta de metadatos ricos puede limitar la comprensión de los resultados del reconocimiento y dificultar el diseño de nuevas tareas de investigación mejoradas.

Hasta este punto, gran parte de los trabajos utilizan una sola fuente de los datos ejecutando sus predicciones sobre uno solo sensor, es decir, de manera unimodal. Lo que se pretende conseguir en este trabajo un sistema de reconocimiento que combine los datos provenientes de distintos sensores además de los de imagen, de manera que se consiga un sistema multimodal, un acercamiento que viene marcando la tendencia en sistemas de HAR recientemente(Badawi et al., 2018; Wen et al., 2022). Se utilizará tanto la problemática planteada como el conjunto de datos del concurso OpenPack Challenge (Yoshimura et al., 2022b). Este *dataset* ofrece una elevada cantidad de horas de registros de diferentes sensores (cámaras, sensores inerciales, láser, cámaras de profundidad) de alta variabilidad

proporcionada por distintos usuarios y alteraciones en las tareas realizadas y enriquecido con unos detallados metadatos. Esto hace que este conjunto de datos sea muy adecuado para resolver este objetivo ya que permite sortear en gran medida los principales desafíos planteados sobre HAR en entornos industriales descritos anteriormente.

Para la resolución de la problemática planteada se pretende profundizar en técnicas de aprendizaje automático, y más en concreto, de aprendizaje profundo. Se va a desarrollar una red neuronal que sea capaz de clasificar las distintas tareas propuesta en el concurso utilizando para ello los datos de los distintos sensores partiendo de los conocimientos adquiridos a lo largo de todo el máster. Se tratará de alcanzar el mejor resultado posible y aprovechar el *leaderboard* de los participantes del concurso original (que finalizó a mediados de enero de 2023) para realizar una valoración comparando con los resultados obtenidos.

1.2. Estructura de la memoria

La memoria de esta proyecto se estructura en los siguientes capítulos:

1. Introducción general, motivación y objetivos.
2. Material: Análisis exploratorio del conjunto de datos que se va a utilizar junto a su estudio e inconvenientes encontrados. En este caso, debido a cómo se desarrolla el trabajo alrededor de este conjunto de datos se ha optado por explicar el contenido de estos antes de observar trabajos previos.
3. Trabajos Previos: análisis de técnicas previas utilizadas en trabajos previos similares y estado del arte.
4. Propuesta y metodología: se formulará una propuesta de solución para la problemática planteada y la metodología a aplicar.
5. Resultados y discusión: presentación de los resultados obtenidos y discusión sobre los mismos, identificando también las limitaciones al trabajo realizado.
6. Conclusiones y trabajos futuros: explicación del aprendizaje realizado junto al planteamiento de posibles mejoras aplicables para futuras iteraciones.

Capítulo 2

Material

En este capítulo se hará un repaso en profundidad del conjunto de datos de OpenPack (Yoshimura et al., 2022b), un *dataset* multimodal que contiene, a partir de datos recogidos por múltiples sensores y cámaras, actividades de operarios realizando tareas de embalaje en un centro logístico.

2.1. Descripción de los datos

OpenPack es el primer conjunto de datos multimodal a gran escala para el reconocimiento de actividades en entornos industriales. Participaron 16 sujetos distintos que empaquetaron un total de 3,956 artículos en 2,048 cajas de envío, y la duración total del conjunto de datos es de 53.8 horas, dividido en 104 sesiones. Se recolectaron un total de 20129 operaciones y 52529 acciones. Hasta la fecha no había un conjunto de datos dan extenso desde LARa (Niemann et al., 2020), que reunía un total de 14.8 horas.

Distingue entre dos tipos de instancias: operaciones y acciones. Las acciones son actividades que comprenden tareas sencillas, secuenciales y unívocas. Las operaciones que corresponden con una tarea más compleja formada por una o más acciones que, de forma conjunta, logran un objetivo en el procesamiento del embalaje. Dentro de este reto nos centraremos en identificar la operación realizada entre las 10 efectivas que son: seleccionar, reubicar etiquetas de artículos, ensamblar cajas, insertar artículos, cerrar la caja, colocar una etiqueta en la caja, escanear la etiqueta, adjuntar una etiqueta de envío, colocar el artículo en un estante y completar una orden. Un listado completo con todas las operaciones, acciones y una breve descripción puede verse en la tabla 2.1.

Tabla 2.1: Listado de operaciones y acciones

Operación	Acción	Descripción	Ignorar
Picking	Pick Up Sheet	Recoger la hoja de pedido de la mesa de trabajo.	

Picking	Pick Up Item from Box	Seleccionar artículos de una caja que contiene elementos recogidos en lotes.	
Picking	Pick Up Box Sheet	Recoger una hoja de pedido y cajas empaquetadas al mismo tiempo. (Esta etiqueta se utiliza para mover cajas empaquetadas y recoger artículos para la siguiente caja al mismo tiempo).	
Picking	Walk to Work Bench	Levantar el artículo y regresar hacia la mesa de trabajo. (Cuando la recogida se hace en un solo viaje, esta acción se establece como el comienzo de la caja).	
Picking	Put Packed Box	Ir a la mesa trasera y colocar la caja empaquetada debajo de la mesa.	
Relocate Item Label	Remove Item Label	Retirar la etiqueta con el código de barras de cada artículo.	
Relocate Item Label	Attach to Order Sheet	Colocar la etiqueta en el margen inferior de la hoja de pedido.	
Relocate Item Label	Hold Pen		
Relocate Item Label	Write Check Mark	El sujeto verifica el nombre y la cantidad de los artículos.	
Relocate Item Label	Put Item Small Bag	Insertar artículos pequeños ($<$10cm) en la bolsa de papel pequeña y cerrarla con cinta. La acción es casi la misma que la de ID0404 pero el momento es diferente.	

Assemble Box	Pick Cardboard	Seleccionar y recoger cajas que coincidan con el tamaño de los artículos.	
Assemble Box	Bend Flap	Doblar las 4 solapas "inferiores" de la caja.	
Assemble Box	Attach Tape	Aplicar cinta adhesiva para cerrar la "parte inferior" de la caja.	
Assemble Box	Turn Over Box	Voltear la caja boca abajo para que el lado con la cinta adhesiva quede en la parte inferior.	
Assemble Box	Pick Up Assembled Box	Recoger una caja pre-ensamblada. (Usado en Escenario 3-5)	
Insert Items	Insert Item into Box	El sujeto agarra los artículos individuales e inserta en la caja ensamblada.	
Insert Items	Air Cushion	Insertar el cojín de aire en la caja.	
Insert Items	Separate Air Cushion	Arrancar el cojín de embalaje conectado.	
Insert Items	Put Item Small Bag	Insertar artículos pequeños ($<$10cm) en la bolsa de papel pequeña y cerrarla con cinta. La acción es casi la misma que la de ID0205 pero el momento es diferente.	
Close Box	Bend Flap	Doblar 4 solapas en la parte superior de la caja.	
Close Box	Attach Tape	Cerrar la caja con cinta adhesiva.	
Attach Box Label	Attach Box Label	Colocar la etiqueta del número de caja en el lateral de la caja.	

Scan Label	Pick Up HT	Recoger el escáner portátil (HT).	
Scan Label	Scan Order Sheet	Escanear el código de ID del pedido en la esquina superior izquierda de la hoja de pedido.	
Scan Label	Scan Box	Escanear la etiqueta del número de caja.	
Scan Label	Scan Item	Escanear cada número de artículo en la etiqueta de la hoja de pedido uno por uno.	
Scan Label	Hold Scanner	Recoger el escáner blanco (para la impresora de etiquetas).	
Scan Label	Scan Order Sheet	Escanear el código de ID del pedido en la esquina superior izquierda de la hoja de pedido con el escáner blanco.	
Scan Label	Scan Printer	Escanear el código en la impresora de etiquetas para imprimir las etiquetas de envío.	
Attach Shipping Label	Pick Up Shipping Label	Recoger la etiqueta de envío impresa de la impresora de etiquetas.	
Attach Shipping Label	Attach Shipping Label	Adjuntar la etiqueta de envío impresa en el lado superior de la caja.	
Put on Back Table	Pick Up Packed Box	Recoger la caja preparada de la mesa de trabajo.	
Put on Back Table	Put Packed Box	Ir a la mesa trasera y colocar la caja empaquetada debajo de la mesa.	
Fill out Order	Pick Up Pen	La mano del sujeto toca la pluma.	

Fill out Order	Write Sign	Firmar cada campo en la columna junto a la lista de artículos en la hoja de pedido.	
Fill out Order	Push Order Sheet into Tray	Insertar la hoja de pedido en la bandeja para hojas de pedido completadas.	
Null	Others		TRUE
Null	System Error		TRUE
Null	Ignore		TRUE
Null	Unknown		TRUE

Durante el embalaje de una caja, el operario tendrá que realizar las operaciones antes descritas en el orden en el que aparecen, sin embargo, las acciones que comprenden el proceso pueden verse alteradas y suprimidas debido a que los paquetes a embalar son distintos en tamaño y forma. Esto afecta también a los datos recogidos por los sensores, pues dependen del tipo de embalaje realizado. A modo de representación, podemos ver cómo se distribuyen a lo largo del tiempo en carácter secuencial en la figura 2.1 en el proceso de embalaje de una caja en una sesión de una operaria a la vez que vemos los distintos datos recogidos por los sensores.

Se escogieron 16 sujetos con distintas condiciones: edad comprendida entre 20 y 50 años, 4 de los sujetos no tenía experiencia previa y los otros 12 entre 1 mes y 4 años, uno de los operadores era zurdo. Se implementó variabilidad en sus condiciones de trabajo mediante diversos escenarios: en ocasiones debían saltarse ciertas acciones o cambiar el orden entre estas, también se introdujo una sirena que se activaba en ocasiones para generar sensación de urgencia y afectar así a los tiempos de operación y forma de actuar, los operadores más experimentados alteraban también el orden para conseguir mayor eficiencia en sus tareas. Se puede ver información completa en el la tabla 2.2.

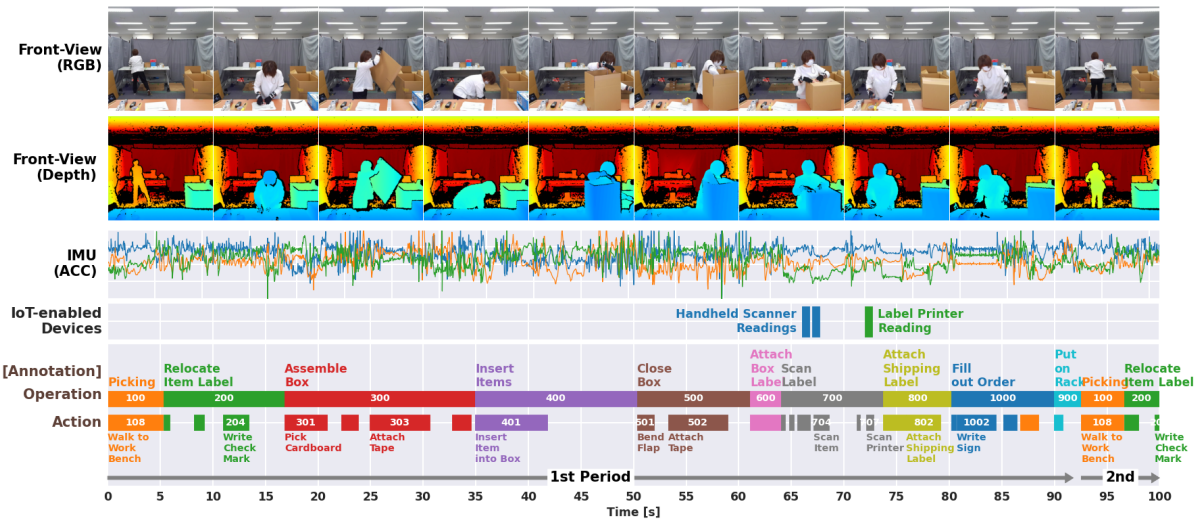


Figura 2.1: Ejemplo de secuencia de embalaje de una caja con múltiples sensores

Sujeto	Sexo	Edad	Mano dominante	Experiencia	Observaciones
U0101	F	-	Derecha	-	
U0102	F	-	Derecha	-	
U0103	F	50s	Derecha	6 Meses	
U0104	F	50s	Derecha	1 Mes	
U0105	F	30s	Derecha	4 Año	
U0106	F	40s	Derecha	1 Mes	
U0107	F	40s	Derecha	3 Años	
U0108	F	30s	Derecha	3 Años	
U0109	M	30s	Izquierda	6 Meses	
U0110	F	40s	Derecha	10 Meses	
U0111	F	50s	Derecha	2 Años	
U0202	F	30s	Derecha	4 Años	Misma persona que U0105
U0203	F	30s	Derecha	3 Años	Misma persona que U0108
U0204	F	40s	Derecha	1 Año	Misma persona que U0110
U0205	F	40s	Derecha	3 Años	Misma persona que U0107
U0207	F	40s	Derecha	20 Meses	
U0210	F	50s	Derecha	3 Meses	Misma persona que U0103

Tabla 2.2: Datos sobre operadores

Los sensores utilizados para la elaboración de este *dataset* son: acelerómetro, giroscopio, cuaternión, pulso de volumen sanguíneo (BVP) y actividad electro-dérmica (EDA), temperatura, keypoints, una nube de puntos LiDAR e imágenes de profundidad. Se utilizó una variedad de sensores, incluyendo unidades IMU en las muñecas y brazos de los sujetos, sensores Empatica E4 en las muñecas para medir BVP y EDA, y cámaras Kinect, LiDAR y RealSense para capturar imágenes y nubes de puntos y elementos IoT como escáneres portátiles e impresoras de etiquetas que emitían señales con su uso. La información completa puede consultarse en la tabla 2.3.

Categoría	Dispositivo	Sensor	Posición	Producto
Visión	kinect	Depth	Vista frontal	Azure Kinect DK
Visión	kinect	Keypoints	Vista frontal	Azure Kinect DK
Visión	rs02	Depth	Vista superior	Intel® RealSense™ Depth Camera D435i
Visión	LiDAR	Point Cloud	Vista frontal	ULTRA PACK(VLP-32C)-Veloidne
Wearable	atr01	Acc/Gyro/ Quaternion	Muñeca derecha	ATR TSND151
Wearable	atr02	Acc/Gyro/ Quaternion	Muñeca izquierda	ATR TSND151
Wearable	atr03	Acc/Gyro/ Quaternion	Brazo superior derecho	ATR TSND151
Wearable	atr04	Acc/Gyro/ Quaternion	Brazo superior izquierdo	ATR TSND151
Wearable	e4-1	Acc/BVP/ EDA/ Temperatura	Muñeca derecha	E4 wristband
Wearable	e4-2	Acc/BVP/ EDA/ Temperatura	Muñeca izquierda	E4 wristband
Dispositivo IoT	Escáner de etiquetas portátil	log	-	8001-C Portable Terminal
Dispositivo IoT	Impresora de etiquetas	log	-	pseudo log data

Tabla 2.3: Sensores de generación del dataset

Por último, se enriqueció toda la información recolectada con abundantes metadatos, indicando identificadores de cada paquete procesado, tamaño, fechas exactas de procesamiento, datos de los operadores, fallos y accidentes ocurridos durante las sesiones o recogidas de datos que se vieron afectadas por problemas de sensores u olvidos.

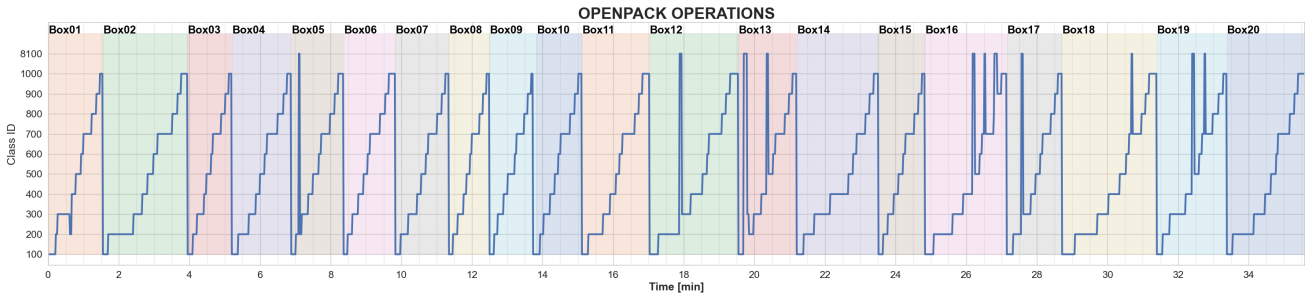


Figura 2.2: Etiquetas de la sesión S0100 del usuario U0101

2.2. Análisis exploratorio

En esta sección se examinará el contenido de los datos para hacernos una idea de los que se maneja. En primer lugar, la imagen RGB no está disponible por motivos éticos y de privacidad de acuerdo a la web oficial[1]. Se ha utilizado jupyter notebook[2] como herramienta para realizar un análisis exploratorio (Cardoso et al., 2019). El cuaderno completo puede verse en el anexo A, donde se han utilizado las facilidades que ofrece la librería OpenPack-Toolkit a la hora de cargar datos mediante configuraciones de Omegaconf[3].

2.2.1. Etiquetas

Para empezar, se han visualizado las etiquetas disponibles. Estas vienen dadas en formato csv tabular donde hay tantas filas como instancias por usuario y sesión con una tasa de una etiqueta por segundo (1Hz) separando en 6 columnas distintas que identifican la marca temporal, usuario, sesión, número de la caja que se está tratando, operación realizada y acción realizada (ver diferencias entre operación y acción 2.1). Una de las herramientas que aporta el *toolkit* son las funciones de *resampling*, que será de gran utilidad a la hora de emparejar en el tiempo datos con las etiquetas, ya que los distintos sensores cuentan con frecuencias de muestro diversas y pueden no corresponder con las de la etiqueta ni entre ellas mismas. En la imagen 2.2 se puede ver el *groundtruth* correspondiente a la sesión S0100 del usuario U0101 donde puede verse la sucesión completa de las distintas operaciones en las 20 cajas tratadas en dicha sesión.

A partir de aquí, los datos de los distintos sensores se irán desglosando por dispositivo.

2.2.2. Sensor IMU

Existen 4 dispositivos ATR-TSND151 ejerciendo de sensores localizados en muñecas y parte superior de ambos brazos. Recogen 3 tipos de datos que son el acelerómetro, el giroscopio y el cuaternión. El cuaternión es un sistema que mediante álgebra permite ubicar la posición absoluta del

[1] https://open-pack.github.io/
[2] https://jupyter.org/
[3] https://omegaconf.readthedocs.io/

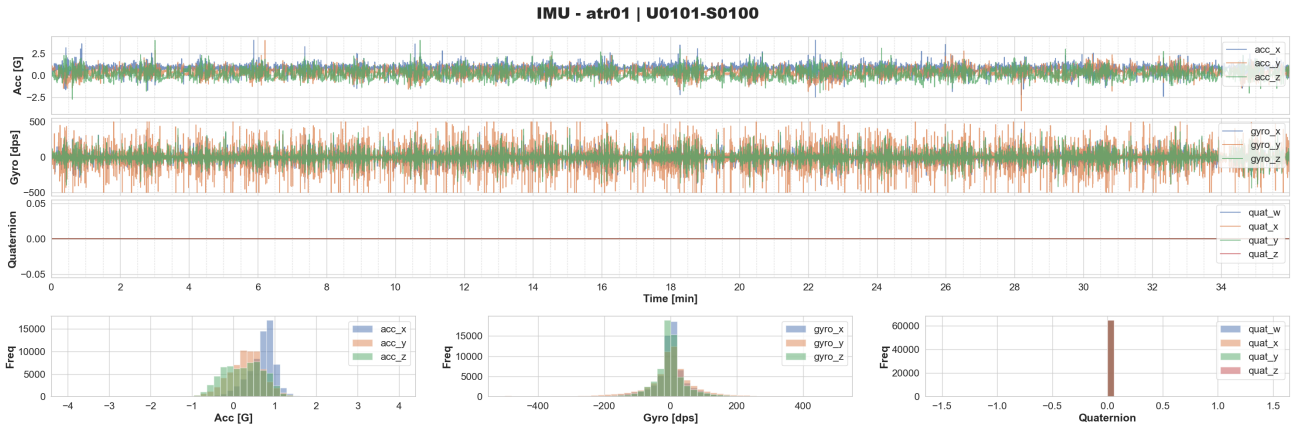


Figura 2.3: Ejemplo de datos del sensor IMU en una sesión

sensor (Bachmann et al., 1999). Por cada uno de estos 3 componentes tendremos valores a lo largo de los ejes x, y, z del acelerómetro medidos en aceleración respecto a g (la aceleración de la gravedad en la tierra) y los del giroscopio, que vendrán en dps (grados por segundo). Sobre cuaternión no se han obtenido datos y vienen prefijados al valor 0. La tasa de muestreo es de 30 Hz, es decir, cada segundo se cuenta con 30 registros de los 3 tipos en un tiempo medido en milisegundos de unix (unixtime).

En la figura 2.3 podemos contemplar estas tres mediciones en sus tres ejes y su distribución, donde se evidencia la falta de datos de cuaternión y los valores de distribución del acelerómetro, que son en su mayoría comprendidos entre -1 y 1 y los del giroscopio que se sitúan entre -200 y 200 aproximadamente pudiendo llegar a rangos mayores en el eje y.

2.2.3. Sensor E4

Se cuenta con dos pulseras de actividad Empatica 4, una en cada muñeca y se recogen 4 tipos de dato distintos: acelerómetro, presión sanguínea, actividad electrodérmica y temperatura. En el tiempo se almacenan también en formato unixtime con una tasa de 32 Hz. En la imagen 2.4 se pueden ver todos ellos donde los datos del acelerómetro se miden en g como en el caso anterior, la presión sanguínea en mmHg, la actividad electro-dérmica en µsiemens y la temperatura en grados celsius. Según las distribuciones puede apreciarse la menor sensibilidad del acelerómetro respecto al previo sensor pese a estar ubicado en una zona similar. También puede verse que los niveles de temperatura y actividad electro-dérmica aumentan en función de la actividad y se mantienen a lo largo de la sesión de trabajo, por lo que no parecen muy ligados a las distintas operaciones.

2.2.4. Sensor Kinect

A partir de este sensor se dan ya extraídos a una tasa de 15 Hz los puntos de articulaciones mediante HRNet utilizando el *framework* MMPose, de OpenMMLab (Sun et al., 2019). De los 17

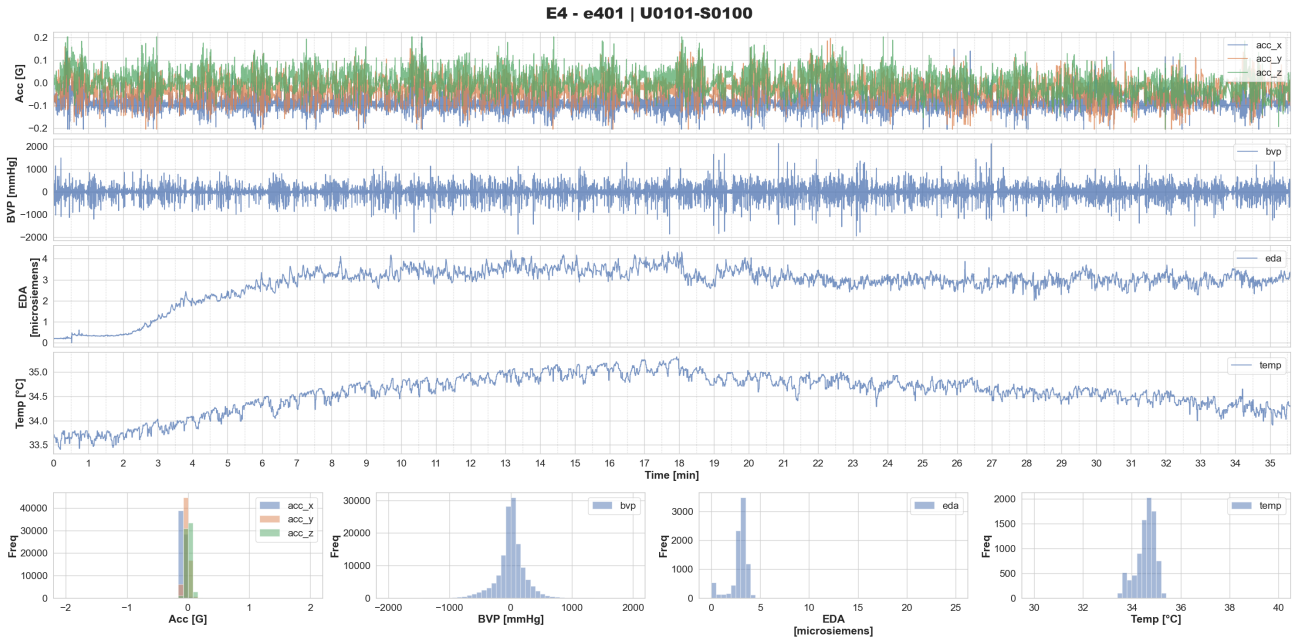


Figura 2.4: Ejemplo de datos del sensor E4 en una sesión

keypoints originales extraemos un total de 15 debido a que los correspondientes a los pies no serán utilizados ya que no son visibles. El uso de la estimación de la pose en el ámbito del reconocimiento de acciones es crucial, como se verá más adelante en el estado del arte y hay que tener en cuenta que los puntos que vienen dados se tratan de predicciones y, por tanto, pueden estar sujetas a fallos o imprecisiones.

A diferencia del resto el formato de estos datos vendrá dado en json, que al ser leído como diccionario python vendrá acompañado de una serie de metadatos para cada usuario y sesión. En la clave info tendremos información básica sobre la sesión, el método utilizado para la estimación de la pose, fecha de creación, etc. y se cuenta también con la etiqueta licencias para indicar el tipo de licencia, que actualmente es Creative Commons 4.0. La etiqueta "categories" contiene la información sobre las instancias detectadas en la escena, que en esta situación siempre va a ser una única persona, y se describe la ubicación de los distintos *keypoints* con sus posiciones de acuerdo a sus identificadores numéricos. Además se incluirá una etiqueta "skeleton" que almacenará todos los pares de ids que tienen conexión (muñeca con codo, codo con hombro, etc.). Por último tenemos la etiqueta "annotations" que almacenará las coordenadas de estos puntos en el marco de la imagen a resolución 1024x1024 y la confianza de dicha predicción indicando también la caja delimitadora o *bounding box* que abarcaría al sujeto.

En la figura 2.5 se pueden ver para un fotograma los puntos correspondientes en rojo con las uniones entre ellos en azul propuestas en la etiqueta "skeleton" mencionada anteriormente.

Haciendo el seguimiento a lo largo de una sesión de dos de esos puntos encontramos algo como lo que puede verse en la figura 2.6, donde se aprecia cierta rutina de repetición que ayudará a determinar la operación realizada en cada momento a lo largo de ambos ejes. También sombreada

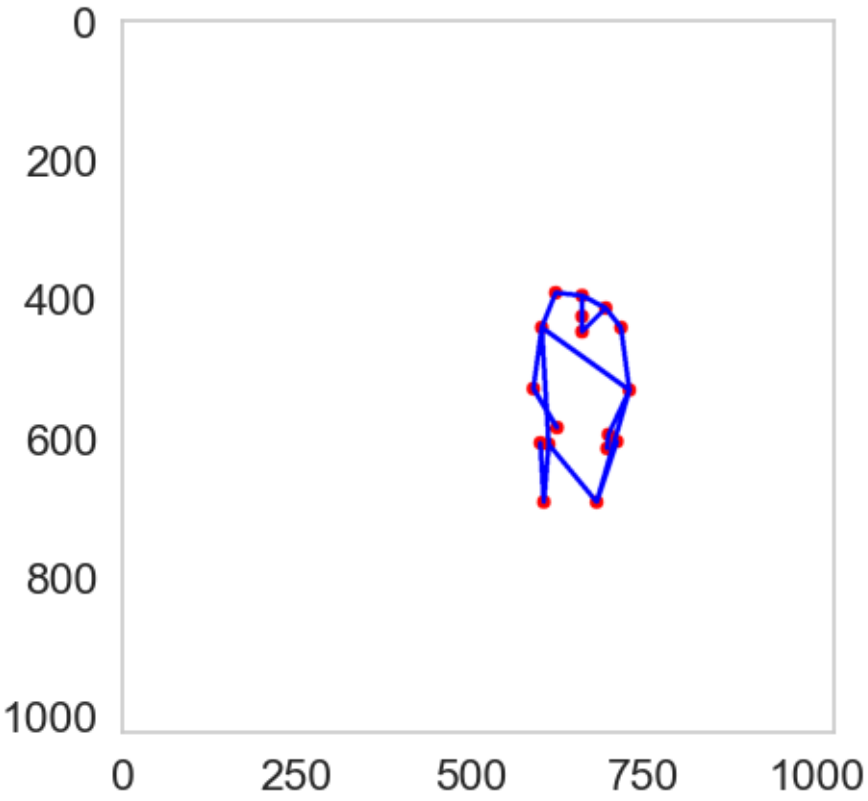


Figura 2.5: Extracción de puntos de pose

puede verse que la confianza de la predicción se mantiene superior a 0.8 de manera constante, lo que indica buena calidad de predicción.

2.2.5. Señales del sistema

A través de instrumentos IoT del trabajador como el lector de etiquetas y la impresora que emiten una señal con su uso (y por lo tanto sin periodicidad) y puede ser utilizada para identificar el momento en el que se produce una determinada operación que involucre dicho uso. Existen también otros datos externos como el listado de pedidos que contiene la información de las cajas, su tamaño

Figura 2.6: Nariz y hombro izquierdo a lo largo de una sesión

Figura 2.7: Muestra de imagen capturada por la cámara de profundidad

y cantidad de elementos que contienen.

2.2.6. Cámara de profundidad

La cámara de profundidad es utilizada habitualmente en sistemas de visión artificial (Lee, 2012) y permite capturar la estructura de la escena evaluando la distancia entre los puntos de dicha escena y la cámara. En este caso se cuenta con una cámara Intel® RealSense™ Depth Camera D435i que es capaz de capturar hasta 90 fotogramas por segundo y se ubica frente al operador. Un ejemplo en escala de grises sobre el que se ha aplicado un reescalado para facilitar su visibilidad puede verse en la figura 2.7 donde la intensidad de cada pixel determina la proximidad siendo los más claros los píxeles que mayor distancia guardan con la cámara.

2.2.7. Sensor LiDAR

Por último, se cuenta también con un sensor LiDAR Velodyne Ultra Puck VLP-32C que recoge datos 3D de la escena en forma de nube de puntos alcanzando frecuencias de captura de 10 Hz y con capacidad de capturar más de 1 millón de puntos por segundo. Las salidas generadas por este sensor han de ser leídas por librerías específicas de nubes de puntos como Open3D en python y teniendo en cuenta que cada sesión almacena gran cantidad de ficheros que superando los 5GB por cada una. Un ejemplo de estas nubes de puntos se puede ver en la figura 2.8 donde se contempla la escena completa y dado que los autores del dataset no han dejado indicaciones puede resultar confuso navegar por él y comprenderlo.

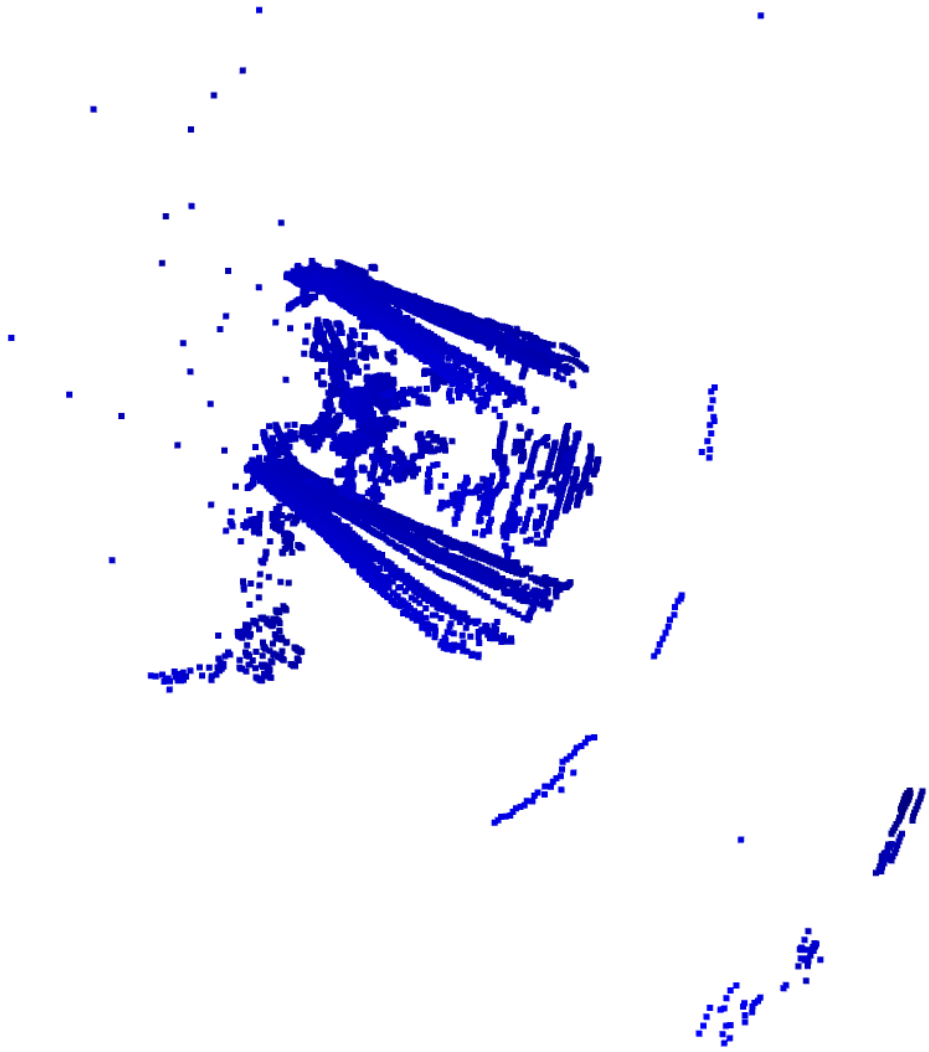


Figura 2.8: Nube de puntos

2.3. Problemáticas encontradas y resolución final

Este dataset contiene una serie de fallos que pueden verse al completo en la tabla 2.4 que principalmente involucran la imposibilidad de utilizar ciertos sensores. Por un lado, algunos de los sensores IMU fueron colocados erróneamente y otros no funcionaron a lo largo de la sesión, se tomarán los datos más completos que se corresponden con el sensor atr01. Las imágenes de cámara de profundidad y las nubes de puntos LiDAR no vienen dadas en el dataset principal y han de ser descargadas aparte [4] además de tener un volumen considerablemente grande para el material con el que se cuenta para realizar este trabajo ya que además requeriría de potencia de computación elevada, por lo que no se utilizarán en este trabajo. Los datos del sensor E4 han sido descartados también debido a redundancia con el IMU y que este segundo además cuenta también con datos de giroscopio, aunque podrán ser tenidos en cuenta y más adelante se discutirá su futuro uso en la sección 6.2. Además de esto existen también ciertos errores en el etiquetado en lo que respecta a tiempos de etiquetado, falta de etiquetas o equivocaciones. Los autores han aclarado que para futuras versiones del dataset (actualmente se encuentra en la versión v0.3.1 con la v1.0.0 en desarrollo) intentarán corregir en la medida de lo posible aquellos fallos enmendables (etiquetado, reemplazo de dispositivos atr03 y atr04 cuando han sido erróneamente intercambiados, etc).

Tabla 2.4: Errores durante la generación del dataset

Sujeto	Sesión	Cajas	Dispositivo	Error	Estado
U0103	S0100	1,2,3,4,5,6, 7,8,9,10,	atr01,atr02, atr03,atr04,	Los sensores ATR no estaban funcionando. El proceso de anotación está suspendido.	PENDIENTE
U0103	S0300	TODAS	atr03,	ATR03 fue colocado en la dirección incorrecta.	NO HAY SOLUCIÓN DISPONI-BLE
U0103	S0400	TODAS	atr03,	ATR03 fue colocado en la dirección incorrecta.	NO HAY SOLUCIÓN DISPONI-BLE

[4] https://github.com/open-pack/openpack-toolkit/issues/7https://github.com/open-pack/openpack-toolkit/issues/11

U0104	S0200	TODAS	atr03,atr04,	La posición de ATR03 y ATR04 parece estar intercambiada.	PENDIENTE
U0104	S0400	TODAS	atr03,atr04,	La posición de ATR03 y ATR04 parece estar intercambiada.	PENDIENTE
U0106	S0400	TODAS	lidar,	Se produjo un error al cargar los datos y el proceso de extracción de cuadros no se ha completado.	PENDIENTE
U0106	S0500	TODAS	lidar,	Se produjo un error al cargar los datos y el proceso de extracción de cuadros no se ha completado.	PENDIENTE
U0107	S0400	TODAS	atr03,	ATR01 fue colocado en la dirección incorrecta.	NO HAY SOLUCIÓN DISPONI-BLE
U0107	S0300	10,11,12, 13,14,15, 16,17,18, 19,20,	kinect,	Faltan los datos de video (Kinect). El proceso de anotación está suspendido.	PENDIENTE

U0108	S0400	1,2,3,4, 5,6,7,8, 9,10,	atr01,atr02, atr03,atr04,	ATR no funcionaba para estas cajas. Los registros de tiempo de las secuencias de datos ATR en el conjunto de datos no son consistentes.	NO HAY SOLUCIÓN DISPONI-BLE
U0109	S0100	17,18, 19,20,	kinect,atr01, atr02,atr03, atr04,	ATR no estaba disponible. El proceso de anotación está suspendido.	PENDIENTE
U0204	S0100	TODAS	atr02,	ATR02 fue colocado en la dirección incorrecta.	NO HAY SOLUCIÓN DISPONI-BLE
U0204	S0500	TODAS	atr02,	ATR02 fue colocado en la dirección incorrecta.	NO HAY SOLUCIÓN DISPONI-BLE
U0207	S0100	TODAS	atr02,	ATR02 fue colocado en la dirección incorrecta.	NO HAY SOLUCIÓN DISPONI-BLE

Debido a las problemáticas mencionadas y dado que el tiempo del que se dispone para realizar este trabajo es limitado, los datos que se tendrán en cuenta para la proposición de un modelo serán del sensor IMU atr01 y atr02 y los *keypoints* de estimación de pose de la estimación del sensor kinect.

Capítulo 3

Trabajos previos

En esta sección se discutirán técnicas utilizadas por otros trabajos similares en el reconocimiento de actividad humana de cara a elaborar un sistema adecuado para los datos con los que contamos en nuestro dataset.

3.1. Inicios de modelos HAR

Las actividades humanas son una serie de tareas continuas compuestas de gestos, acciones e interacciones individuales o múltiples. El gesto se refiere al movimiento de partes del cuerpo para enfatizar el habla, mientras que la acción se refiere al movimiento colectivo de partes del cuerpo para completar una tarea. Por ejemplo, mover la cabeza en negación es un gesto, caminar es una acción y hablar en voz alta con expresiones faciales desagradables es un comportamiento enojado. La interacción es una colección de acciones generalmente realizadas por dos o más sujetos, por ejemplo, una conversación entre dos personas, pelea, cocinar, entrada de datos o lavado de autos, etc. Las actividades en grupo son realizadas por varias personas y pueden incluir una colección de gestos, acciones e interacciones, por ejemplo, jugar al fútbol o una manifestación. Los gestos y acciones son fáciles de reconocer y se consideran simples, mientras que el comportamiento y las interacciones son intermedios. Las actividades con varias personas, como la interacción humano-humano, grupales o los eventos, son altamente complejas. Considerando las subactividades mencionadas anteriormente, los enfoques utilizados para reconocerlas varían ampliamente. Algunos métodos básicos incluyen técnicas de procesamiento de imágenes basadas en características, sustracción de fondo/figura, detección y clasificación de acciones (por ejemplo, flujo óptico, puntos de interés espacio-temporales)(Huang, 2010; Cheng et al., 2010; Oral and Deniz, 2007).

Los métodos avanzados son una combinación de múltiples pasos, que pueden extraer colectivamente características avanzadas y realizar un análisis en profundidad para reconocer actividades humanas (Yilmaz et al., 2004; Bernardin and Stiefelhagen, 2008; Bolme et al., 2010; Cucchiara et al., 2003). La mayoría de los sistemas estaban basados en modelos probabilísticos, descriptores, procesos

estocásticos y técnicas de aprendizaje automático como SVM o SOM) (Vrigkas et al., 2015)y otros métodos básicos basados en visión artificial, como el flujo óptico (Denman et al., 2009; Ince and Konrad, 2008; Morris and Trivedi, 2008), los puntos de interés espacio-temporales (STIP) (Laptev, 2005), los modelos ocultos de Markov (HMM) (Blunsom, 2004). Conforme las tecnologías fueron evolucionando, surgieron trabajos como el de (Donahue et al., 2015) donde se aplicaban herramientas avanzadas de aprendizaje profundo o *deep learning*, por ejemplo, redes neuronales convolucionales (CNN), redes neuronales recurrentes (RNN) (Nunez et al., 2018; Chen et al., 2017; Li et al., 2017) y también en (Simonyan and Zisserman, 2014; Wang et al., 2015, 2016; Sun et al., 2018b). Tras la evolución de estas, se trasladó al dominio de convoluciones 3D en (Carreira and Zisserman, 2017; Tran et al., 2015).

El HAR tiene una naturaleza multidisciplinaria y afecta a diversos sistemas de la vida cotidiana y desempeña un papel en entornos interiores/exteriores, robótica, recuperación de información basada en contenido, interacciones humano-máquina, vigilancia de seguridad, videovigilancia, sector educativo, monitoreo y aplicaciones basadas en interacción social (Kerber et al., 2017). Teniendo en cuenta los trabajos anteriores, los sistemas HAR se pueden clasificar como en línea u fuera de línea según los datos de entrada y la estrategia de procesamiento. Luego, existen enfoques unimodales/multimodales que utilizan diferentes modalidades, como cuadros de video, pistas de audio, datos de esqueleto y datos de profundidad. La mayoría de los estudios anteriores han discutido enfoques fijados (*handcrafted*), y algunos trabajos recientes también han incorporado enfoques basados en aprendizaje (Vishwakarma and Agrawal, 2013; Zhen et al., 2016; Sargano et al., 2017).

Dentro de estos estudios tenemos por ejemplo (Gavrila, 1999), que separó la investigación en enfoques 2D (con y sin modelos de forma explícitos) y enfoques 3D. En (Aggarwal and Ryoo, 2011), se presentó una nueva taxonomía centrada en el análisis de movimiento humano, el seguimiento desde cámaras de vista única y múltiple, y el reconocimiento de actividades humanas. Similar en espíritu a la taxonomía anterior, (Wang et al., 2013) propusieron una jerarquía de categorización de acciones jerárquicas. El trabajo de (Moeslund et al., 2006) se centró principalmente en métodos de reconocimiento de acciones basados en poses y propuso una taxonomía de cuatro niveles, que incluye la inicialización del movimiento humano, el seguimiento, la estimación de poses y los métodos de reconocimiento.

(Turaga et al., 2008) propusieron una clara distinción entre los significados de "acción" y "actividad", donde los métodos de reconocimiento de actividades son de mayor complejidad que las acciones y se categorizaban según su grado de complejidad de actividad. (Poppe, 2010) caracterizó los métodos de reconocimiento de actividades humanas en dos categorías principales, describiéndolos como "arriba hacia abajo" y "abajo hacia arriba". Por otro lado, (Aggarwal and Ryoo, 2011) presentaron una taxonomía estructurada en árbol, donde los métodos de reconocimiento de actividades humanas se categorizaron en dos subcategorías principales: enfoques de "una sola capa" y enfoques "jerárquicos", cada uno de los cuales tiene varias capas de categorización.

Además de imagen, en la sensórica también ha habido gran cantidad de estudios como (Pham

et al., 2020; Kalabakov et al., 2021)que mediante acelerómetros y giroscopios son capaces reconocer actividades a cambio de necesitar capturar los movimientos a través de un dispositivo específico. Partiendo de estos sensores, añadir la imagen ha generado modelos multi-modales que cuentan con un creciente interés en los últimos años. (Wei and Kehtarnavaz, 2020; Wei et al., 2019, 2020) utilizan combinaciones de CNN (redes neuronales convolucionales) 3D para imagen con CNN en 2D para sensores inerciales. En (Diete and Stuckenschmidt, 2019; Diete et al., 2019)utilizan un reloj inteligente para medir los datos inerciales y unas gafas inteligentes para grabar el vídeo de las acciones con visión egocéntrica (1ª persona), donde los datos inerciales capturaban los movimientos de antebrazos y a partir de la visión los objetos y se realiza una técnica de fusión en la que se juntan los datos provenientes de los distintos sensores con los de imagen. Las técnicas de fusión para formar un único vector característico se dividen dependiendo de si se producen al inicio del reconocimiento (fusión temprana o *early fusion*) o tras haber pasado por mecanismos de reconocimiento individuales y juntarse al final de estos (fusión tardía o *late fusion*), que es precisamente la usada en ambos casos mencionados. De hecho, en el propio trabajo sobre el dataset (Yoshimura et al., 2022b)se proponen diversos métodos que utilizan estas distintas estrategias.

3.2. Reconocimiento de actividades humanas con sensores inerciales

En el dominio de la sensórica mediante *wearables,* los algoritmos de *deep learning* son altamente capaces y eficientes en el manejo de señales de series temporales para extraer características y realizar clasificaciones. Esto ha atraído a los investigadores a aplicar técnicas de deep learning, como CNN, LSTM y modelos híbridos, en aplicaciones de reconocimiento de actividades humanas.

3.2.1. Modelos basados en CNN

Los modelos de Redes Neuronales Convolucionales (CNN) son lo suficientemente versátiles como para clasificar imágenes y predecir señales de series temporales. Esta ventaja ha llevado a varios investigadores a integrar modelos de CNN para clasificar señales inerciales en de sensores para el reconocimiento de actividades humanas. La extracción de características mediante CNN tiene ciertas ventajas sobre las técnicas tradicionales de aprendizaje superficial en el campo del HAR, como la dependencia local, la invariancia a la escala y la capacidad para capturar todas las posibles interacciones complejas no lineales entre las características (Wang et al., 2019a).

(Chen and Xue, 2015) propusieron un modelo de CNN para reconocer actividades simples a través de señales de acelerómetro triaxial de un teléfono inteligente. El modelo de CNN fue construído y los núcleos de convolución se modificaron para procesar las señales del acelerómetro triaxial. (Ronao and Cho, 2016) propusieron una arquitectura de CNN novedosa para extraer características complejas

utilizando una capa de convolución explotada (1×9-1×14) con un tamaño de *pooling* bajo (1×2-1×3). El enfoque propuesto consistía en la CNN construida con señales en bruto y características temporales de señales transformadas de Fourier rápida para añadir información adicional.

(Ignatov, 2018) propuso una CNN para la extracción de características locales junto con características estadísticas como la media, la varianza, la suma de valores absolutos y el histograma de cada entrada para preservar información sobre características globales de las señales. Este trabajo de investigación también comprueba el impacto de la longitud de la señal al variar la ventana deslizante hasta 1 segundo en el rendimiento. (Avilés-Cruz et al., 2019) desarrolló un marco de aprendizaje profundo que construye el modelo de CNN para clasificar y analizar el reconocimiento de actividades exclusivamente dependientes del usuario. En este enfoque, se construyen tres modelos de CNN (fino, medio y grueso) en paralelo para la extracción de características locales y se combinan en un solo modelo de clasificación. Los modelos de CNN generalmente requieren recursos computacionales elevados para la extracción de características y la clasificación, ya que utilizan un mayor número de filtros y mapeos. Para reducir el costo del consumo de energía y requisitos de hardware, (Wan et al., 2020) diseñó un modelo de CNN para la extracción de características locales de la señal del acelerómetro de tres ejes de un teléfono móvil siendo capaz de ejecutarse desde el propio teléfono. Se utilizaron los datasets PAMAP2 y HAR para comparar con LSTM, BLSTM, MLP y SVM.

Para reducir el coste de memoria y computacional de la CNN tradicional, (Tang et al., 2020) ha utilizado filtros Lego en lugar de filtros de convolución. El modelo ligero propuesto funciona y escala mejor ya que no requiere una estructura de red especial ni recursos computacionales. Del mismo modo, (Cheng et al., 2022) introdujo una nueva CNN eficiente en cálculos utilizando una convolución parametrizada condicional para el HAR en tiempo real en dispositivos móviles y *wearables*. Se demostró la eficiencia de la red desarrollada con capacidad de gran escala en comparación con el modelo de referencia.

Además de los recursos informáticos elevados, la CNN es más propensa a un problema de inicio en frío. Para superar este problema, el trabajo de investigación (Cruciani et al., 2020) evaluó varios modelos preentrenados de CNN utilizando actividades complejas en tiempo real. Además, todos los modelos preentrenados fueron evaluados en diferentes hiperparámetros para identificar los mejores modelos de CNN para el HAR. Luego, el modelo candidato identificado se implementó como un modelo extractor de características para evaluar un conjunto de datos del mundo real a gran escala.

Para desarrollar un sistema HAR más eficiente y en tiempo real, (Nutter et al., 2018) presentaron un marco de aprendizaje profundo que reconoce actividades en tiempo real a través del propio teléfono móvil. Las características tradicionales hechas a mano de los datos IMU se reducen dimensionalmente mediante el Análisis de Componentes Principales (PCA) a 100 características. Esto minimiza el tamaño del modelo en las características de entrenamiento y prueba y reduce la sobrecarga computacional para ejecutarse en un procesador incorporado, lo que preserva la vida útil de la batería de un teléfono inteligente. El modelo propuesto introdujo una SVM híbrida de dos clases en lugar de una capa densa para mejorar aún más el rendimiento de la clasificación.

Los modelos de CNN propuestos son altamente competentes, eficientes y logran una mayor precisión en el reconocimiento de actividades humanas simples. Sin embargo, en actividades más complejas y confusas que puede realizar cualquier persona mayor en tiempo real, como buscar debajo de una cama o atarse los cordones de los zapatos, la precisión puede fluctuar. Para reducir las fluctuaciones en la tasa de precisión y mejorar la sensibilidad y la especificidad en el reconocimiento de HAR confuso, (Zhu et al., 2019) presentó un conjunto de CNN con capas y filtros variables. El modelo propuesto reconoce las actividades confusas y el movimiento dinámico de las personas en actividades con menos datos de entrenamiento. Para investigar el sistema de reconocimiento de actividades humanas, (Khan et al., 2018) propuso un modelo de transferencia de aprendizaje transductivo llamado Heterogeneous Deep CNN (HDCNN). Este modelo HDCNN construye una CNN de dos capas con divergencia de Kullback-Leibler para ajustar las propiedades de la CNN. El modelo propuesto asigna una distribución invariante de pesos a las capas de la CNN, siempre que se esté monitoreando el proceso de actividades. Se entrenó el modelo con señales de un teléfono y para su evaluación se usaron señales de reloj inteligente y viceversa para el reconocimiento de actividades. Debido a su capacidad de adaptación entre dominios, el modelo propuesto muestra un rendimiento óptimo en comparación con el entrenamiento y test convencionales. Además de clasificar actividades cotidianas simples y repetitivas, (Xiao et al., 2021) propuso un modelo de aprendizaje profundo con una función de penalización no supervisada para reconocer actividades complejas. En el sistema propuesto, se utilizó un dataset simulado de AMASS para el entrenamiento fuera de línea para aumentar la variedad y diversidad. El conjunto de datos contiene una amplia variedad de poses humanas y datos IMU virtuales. Durante las pruebas, el sistema propuesto integraba un mecanismo de aprendizaje por transferencia para mejorar el rendimiento ajustando finamente la red neuronal parcial.

Aunque los modelos de CNN muestran un buen rendimiento en el HAR, todavía enfrentan ciertos desafíos para lograr una precisión óptima además de requerir grandes cantidades de recursos computacionales.

3.2.2. Modelos basados en LSTM

Los modelos LSTM son altamente eficaces para predecir secuencias de señales de series temporales. Mientras que los modelos CNN explotan "correlaciones espaciales" para clasificar imágenes, Las LSTM procesan secuencias completas de datos a través de conexiones de retroalimentación para clasificar datos de series temporales. Dado el esfuerzo ventajoso de LSTM sobre el modelo CNN (Ronao and Cho, 2016), los investigadores han propuesto ciertas técnicas en modelos de HAR basados en LSTM.

(Chung et al., 2019) realizaron un estudio piloto con 15 participantes sobre el sistema de posicionamiento de sensores en el cuerpo. Este estudio crea un banco de pruebas para varias actividades simples utilizando ocho sensores inerciales IMU llevados en el cuerpo. Luego, se implementó el modelo

LSTM para evaluar el rendimiento en entornos del mundo real y en un banco de pruebas controlado. Además, se integró un modelo de conjunto (*ensemble*) con el modelo LSTM para demostrar la probabilidad de clase de la modalidad de múltiples sensores. El sistema propuesto tiene ciertos desafíos, ya que creó un banco de pruebas para el reconocimiento de actividades simples con menos participantes y, por lo tanto, no es adecuado para aplicaciones a gran escala.

Para diseñar un HAR con menos potencia de cálculo y latencia reducida, (Agarwal and Alam, 2020) propusieron un modelo de aprendizaje profundo ligero. El modelo propuesto es adecuado para implementarse directamente en el dispositivo (*edge computing*), ya que se reduce la latencia de comunicación y se puede aplicar en escenarios reales. Sin embargo, este sistema no evalúa su rendimiento con actividades más complejas antes de ser llevado a la implementación en tiempo real en pacientes ancianos. (Rashid et al., 2022) han ampliado (Agarwal and Alam, 2020) y propuesto una CNN adaptable eficiente en energía y memoria para dispositivos de borde de bajo consumo energético. Se llevó a cabo una experimentación con actividades complejas para validar la importancia del sistema propuesto en términos de eficiencia en memoria y energía utilizando los datasets de opportunity y w-HAR. (Zhao et al., 2018) han propuesto una arquitectura de LSTM residual bidireccional con la ventaja de concatenar el estado hacia adelante y el estado hacia atrás, es decir, dirección de tiempo positivo y negativo. La conexión residual utilizada entre las celdas apiladas evita el problema de desvanecimiento del gradiente.

Los sistemas propuestos (Agarwal and Alam, 2020; Rashid et al., 2022; Zhao et al., 2018) tienen un rendimiento relativamente bajo en el reconocimiento de actividades humanas complejas, dinámicas y posturales. Para resolver este problema, (Wang and Liu, 2020) propusieron una nueva estructura de aprendizaje profundo llamada Hierarchical deep Long-Short Term Memory (H-LSTM). Las señales sensoriales se preprocesan para eliminar el ruido y las distorsiones, y luego el modelo H-LSTM extrae características en el dominio tiempo-frecuencia para mejorar el rendimiento del sistema. Para lidiar con datos de series temporales en bruto,(Ashry et al., 2020) propusieron un modelo LSTM en cascada (CHARM), un modelo *online* y *offline* para abordar la necesidad de un gran conjunto de datos de entrenamiento utilizando nuevas características hechas a mano como se describe en (Ashry et al., 2018; Gomaa et al., 2017), junto con señales.

En su mayoría, un sistema HAR se ha resuelto tradicionalmente utilizando características locales extraídas con métodos heurísticos. Los algoritmos de aprendizaje automático tradicionales pueden caer fácilmente en mínimos locales en lugar de una solución óptima global. El trabajo de investigación (Sun et al., 2018a) propuso un marco híbrido profundo basado en operaciones de convolución de CNN integrado con unidades recurrentes LSTM y clasificado mediante ELM (*Extreme-Learning Machine*) para superar este problema. El marco propuesto se evaluó en el dataset opportunity y supera a los modelos de redes neuronales no recurrentes.

Otro problema importante que deben abordar los modelos LSTM en HAR es analizar los datos de sensores con datos débilmente etiquetados. (Zhou et al., 2020) han diseñado un marco de aprendizaje semisupervisado utilizando LSTM con Deep Q-Network y un esquema de autoetiquetado inteligente

para mejorar el rendimiento en datos de sensores etiquetados débilmente. Deep Q-Network resuelve mejor las muestras etiquetadas de manera insuficiente o débil para mejorar el rendimiento del sistema. El modelo LSTM en el sistema propuesto reconoce los patrones detallados extraídos contextualmente de los datos de movimiento secuenciales.

3.2.3. Modelos híbridos

Los modelos híbridos fueron ideados para funcionar mejor que los modelos tradicionales de aprendizaje profundo en ciertos desafíos, como el entrenamiento limitado, actividades confusas y la localización de sensores. (Xia et al., 2020a) propusieron un modelo híbrido profundo integrando modelos LSTM conectados con capas de convolución y capas de agrupación global (GAP). Se utiliza GAP en lugar de una capa completamente conectada tradicional. Además, se introdujo la *batch normalization* después de GAP para acelerar la convergencia del sistema propuesto.

La mayoría de los sistemas HAR se proponen para actividades simples y presentan ciertos desafíos en el reconocimiento de actividades físicas complejas. Para monitorear y detectar actividades dinámicas y complejas de sujetos humanos, (Qi et al., 2020) propusieron un marco de reconocimiento adaptativo. El marco propuesto introdujo un aprendizaje en línea independiente de las restricciones de clase y clasificó 12 clases de actividades. Se realizó una experimentación para mostrar que el marco propuesto logra la mayor precisión del 95.15 % y el 92.20 % en la detección de 5 dinámicas, 6 estáticas y secuencias de transiciones cuando los teléfonos móviles se mantienen en la cintura y el bolsillo. Además de las actividades complejas para reconocer actividades de transición de corta y larga duración, el trabajo de investigación (Wang et al., 2020) propuso un esquema de aprendizaje profundo. El modelo de sensor portátil combinado reconoce con precisión las actividades y sus transiciones.

Sistemas de reconocimiento de acciones como (Qi et al., 2020; Wang et al., 2020), enfrentan ciertos desafíos, en particular el reconocimiento de actividades similares y confusas, como caminar, subir escaleras, etc., con menos dispersión interclase y alta dispersión intraclase. Para superar este desafío, (Lv et al., 2020) introdujeron un mecanismo de margen para extraer características discriminativas para la clasificación. Se modificaron cuatro redes neuronales utilizando el mecanismo de margen propuesto, y el rendimiento se comparó con modelos tradicionales en varios conjuntos de datos de referencia.

El HAR basado en *smartphones* tiene otros desafíos, como la ubicación del dispositivo y la dependencia del sujeto, que producen una alta tasa de falsas alarmas y una baja tasa de alarmas positivas. Esto se debe principalmente a que proporciona información sobre una clase de acción y no sobre el contexto del usuario, como la ubicación y la manipulación de objetos. Para abordar este desafío, el trabajo de investigación (Atienza, 2019) integró modelos CNN y LSTM. Los resultados demostraron que los modelos CNN y LSTM funcionaron bien en reconocimientos independientes de la ubicación y el sujeto. El trabajo de investigación (Rueda et al., 2019) también propuso un

modelo de reconocimiento de actividad híbrido para abordar la dependencia de la ubicación y el sujeto que combina un modelo de red neuronal profunda con un modelo simbólico. Se realizó una experimentación exhaustiva en la preparación de sopa de zanahoria para analizar el rendimiento propuesto en información compleja y en un contexto realista.

(Mukherjee et al., 2020) propusieron EnsemConvNet, un conjunto de modelos CNN-Net, Encoded-Net y CNN-LSTM que utiliza arquitecturas de aprendizaje profundo con características de aprendizaje superficial y estadísticas tradicionales. Lo propuesto utiliza un enfoque ponderado automatizado que incluye votación mayoritaria, fusión de puntajes, regla de suma y regla de producto para identificar las combinaciones de clasificadores. Se evaluaron los rendimientos sobre los datasets WISDM, UniMB SHAR y MobiAct.

(Su et al., 2019) integraron el modelo Deep Bidirectional Long Short-Term Memory (Deep DBLSTM) y el modelo CNN para un sistema HAR automatizado. La larga secuencia de datos en bruto se procesa utilizando el modelo DBLSTM para generar un vector de salida bidireccional. Luego, se utiliza CNN para extraer características locales del vector de salida. Finalmente, la red completamente conectada con función softmax clasifica las actividades humanas.

La atracción reciente hacia el desarrollo de técnicas de aprendizaje profundo robustas en la atención médica inteligente para tomar decisiones utilizando datos sensoriales multimodales ha aumentado significativamente. (Gumaei et al., 2019) propusieron un modelo efectivo de reconocimiento de actividades basado en sensores multimodales utilizando un modelo híbrido de aprendizaje profundo. El modelo integra unidades recurrentes simples (SRU) y unidades recurrentes con compuertas (RGUs) de redes neuronales. Las SRU profundas se utilizan para procesar las secuencias de entrada multimodales utilizando su capacidad de estado de memoria interna. Además, se utiliza RGUs profundas para su mecanismo de retroalimentación para mejorar el rendimiento de la estabilidad en el reconocimiento de actividades. (Xu et al., 2019) propusieron una combinación del modelo de red neuronal Inception y el modelo de red neuronal recurrente para reconocer entradas de sensores multicanal. El módulo Inception con diversas capas de convolución basadas en kernel extrae las características multidimensionales. Finalmente, la clasificación se realizó utilizando una capa GRU.

Como se mencionó anteriormente en las ventajas de las técnicas de aprendizaje profundo, varios trabajos de investigación (Wang et al., 2021; He et al., 2018; Wang et al., 2019b) propusieron mecanismos de atención para funcionar bien incluso en tareas de reconocimiento de actividades débilmente etiquetadas y tareas de localidad. (He et al., 2018) propusieron un HAR supervisado débilmente basado en el aprendizaje de atención recurrente que utiliza el aprendizaje por refuerzo para interactuar con los datos del sensor a lo largo del tiempo. Inicialmente, el contexto y las características locales representativas de CNN se extraen a través de varias iteraciones aplicando un nuevo sistema de recompensas introducido a través del aprendizaje de atención recurrente. Dado que la etiquetación de la colección de datos es más difícil para secuencias de actividades complejas y más largas, (Zhu et al., 2018) propusieron un modelo de aprendizaje profundo semi-supervisado utilizando el ensamblaje temporal de la memoria LSTM.

(Tang et al., 2021) propusieron SelfHAR, un modelo semi-supervisado que aprende eficazmente a aprovechar conjuntos de datos de detección móvil no etiquetados para complementar conjuntos de datos etiquetados pequeños. Este enfoque combina el autoentrenamiento junto con la ampliación con el modelo CNN. (Bhat et al., 2020) introdujeron redes de atención recurrente (RAN) que combinan el mecanismo de atención con la extracción de características de CNN y un decodificador LSTM. El RAN propuesto por (Bhat et al., 2020) a menudo tiene una característica débil en comparación con el CNN tradicional. Además, el RAN se concentra en la atención dura y suave, que presta más atención a la actividad objetivo durante una secuencia larga y no aborda las dependencias espacio-temporales de las señales de detección multimodales. Para superar esto, (Gao et al., 2021) propusieron DanHAR: una nueva atención dual (combinando atención de canal y atención temporal) en CNN para demostrar la superioridad en la mejora de la comprensibilidad de HAR multimodal. (Wang et al., 2019b) propusieron un nuevo HAR basado en la atención que modifica el CNN tradicional mediante un mecanismo de atención. El modelo propuesto calcula la compatibilidad entre características globales y locales extraídas en las capas completamente convolucionales finales y las capas convolucionales, respectivamente.

El HAR es principalmente importante en servicios conscientes del usuario y del contexto que deben monitorearse continuamente en la vida diaria. En general, las actividades humanas tienen un período de tiempo más largo que el sistema HAR, que reconoce durante unos segundos. Por lo tanto, el monitoreo continuo de las actividades humanas utilizando estos sistemas propuestos es ineficiente desde el punto de vista computacional. Para abordar la detección de cambios a nivel de segmento consciente del contexto, (Jeong and Kim, 2019) propusieron un mecanismo de detección de cambios de bajo costo computacional para reconocer las actividades. En la propuesta, se ha introducido una Red Neural Convolucional Completa (FCNN) con una alta tasa de reconocimiento para clasificar las actividades. Supera a las CNN tradicionales y otras técnicas convencionales en una experimentación extensa. Además, el sistema propuesto consume 6.5 veces menos de energía que los modelos convencionales en plataformas integradas.

A pesar de que CNN, LSTM y el modelo híbrido capturan características espacio-temporales, (Singh et al., 2020) propusieron una capa de autoatención para centrarse en información específica de las incrustaciones generadas por cualquier modelo de aprendizaje profundo. Aquí, se utilizó la capa de autoatención para aprender pesos que capturan las relaciones latentes entre los puntos de tiempo de entrada de datos sensoriales crudos para decodificar la actividad humana de manera eficiente. Se realizó una experimentación en seis HAR diferentes para validar la importancia de la capa de autoatención.

(Ascioglu and Senol, 2020) abordaron el reconocimiento de actividades en entornos exteriores. Han generado un nuevo conjunto de datos mediante un novedoso sistema de monitoreo de actividades inalámbrico basado en sensores y sus aplicaciones en redes neuronales de aprendizaje profundo. La experimentación se llevó a cabo a través de CNN, LSTM, ConvLSTM. (Ordóñez and Roggen, 2016) propusieron un modelo DeepConvLSTM basado en el éxito de las redes neuronales recurrentes para

reconocer actividades multimodales y modelar explícitamente la dinámica temporal de las activaciones de características y será explicado más adelante.

3.3. Reconocimiento de actividades humanas en imagen

En paralelo en el dominio de la imagen, la aparición de modelos que utilizaban la pose o esqueleto hacia el año 2014 donde (Lillo et al., 2014; Toshev and Szegedy, 2014; Wei et al., 2016) marcaron el camino consiguiendo resultados cada vez mejores, los primeros modelos de *deep learning* que aparecieron estaban basados en redes recurrentes (Du et al., 2017) y continuaron desarrollándose a través de mapas convolucionales con componentes temporales como en (Wei et al., 2016). Con el tiempo, (Liu and Yuan, 2018; Xu et al., 2018; Zhang et al., 2019a; Li et al., 2020b) perfeccionaron las técnicas de reconocimiento de actividad utilizando la pose y superó aquellos modelos que utilizaban el convoluciones 3D. Estos métodos han ganado importancia sobre otras modalidades porque contienen lo siguiente:

1. Información espacial, que implica una fuerte correlación entre el nodo conjunto y sus nodos adyacentes para modelar información estructural intrínseca en un fotograma.

2. Información temporal, que modela una fuerte correlación temporal dentro de los fotogramas de manera intrafotograma, adicionalmente.

3. La relación de co-ocurrencia se modela de manera elegante utilizando la modalidad del esqueleto cuando se tienen en cuenta simultáneamente los dominios espacial y temporal.

Las redes neuronales convolucionales basadas en grafos han logrado un gran avance en los últimos años. Existe una similitud entre el esqueleto del cuerpo humano y un grafo; por lo tanto, las redes convolucionales de grafos se han utilizado ampliamente para el reconocimiento de acciones basado en el esqueleto. En el siguiente apartado se resumirán las técnicas recientes de reconocimiento de acciones basadas en grafos.

3.3.1. Redes convolucionales de grafos

Las redes neuronales convolucionales se extendendieron al espacio no euclídeo utilizando la red convolucional basada en grafos (GCN), como se propuso en (Kipf and Welling, 2016). Para comprender el funcionamiento de estas de manera gráfica puede observarse la figura 3.1, donde puede verse que los elementos de la convolución dependen de sus relaciones y no de su proximidad espacial. Al aplicar esta idea a datos no euclídeos, los investigadores han desarrollado el uso de GCN para el reconocimiento de acciones basado en el esqueleto. El reconocimiento de acciones basado en grafos implica el uso de GCN para reconocer diferentes acciones, y el esqueleto del cuerpo es la modalidad de entrada para este tipo de arquitecturas. En el trabajo pionero para modelar el reconocimiento

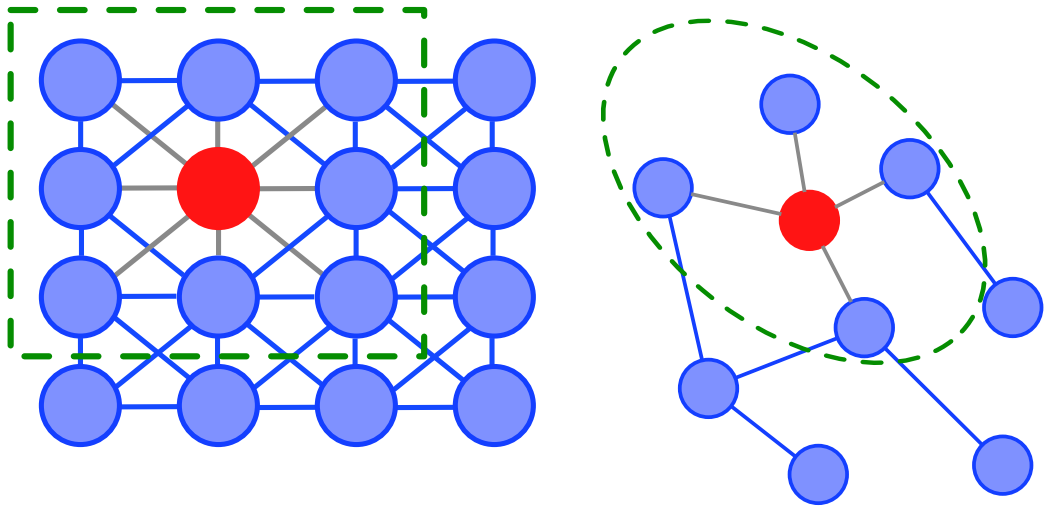


Figura 3.1: Muestra de CNN (izquierda) y GCN (derecha)

de acciones basado en el esqueleto utilizando GCN (Yan et al., 2018), los esqueletos del cuerpo se representan como grafos espacio-temporales a lo largo de la duración de una secuencia de video. Este concepto se ilustra en la figura 3.2, donde se pueden apreciar cuatro representaciones distintas de grafos del esqueleto de cara a tratarlos con convoluciones. Durante los últimos cinco años, han surgido numerosos enfoques para el reconocimiento de acciones basado en el esqueleto utilizando GCNs.

3.3.1.1. Topologías

Ha habido muchos trabajos sobre el desarrollo de las GCN. El primer artículo realizado por (Bronstein et al., 2017), se centró principalmente en la operación de convolución en grafos, así como en su importancia y aplicaciones. (Hamilton et al., 2017) investigaron el aprendizaje de representaciones e incrustaciones de redes en grafos. (Lee et al., 2019) llevaron a cabo un estudio exhaustivo sobre las redes de atención basadas en grafos. Otro trabajo con una nueva taxonomía para GCN fue propuesta por (Wu et al., 2020). (Zhang et al., 2019b) realizaron una estudio completo de técnicas GCN desarrolladas recientemente. (Zhang et al., 2020) categorizaron los métodos de convolución de grafos en cinco subclases: redes neuronales recurrentes de grafos (RNN), GCN, *autoencoders* de grafos, aprendizaje por refuerzo en grafos y métodos adversarios en grafos. Esta literatura mencionada anteriormente destacó los desarrollos recientes en arquitecturas GCN generales y abogó por sus aplicaciones en diferentes áreas. La taxonomía que utilizaremos en este trabajo será la propuesta por (Ahmad et al., 2021) donde contamos con las siguientes 5 categorías:

1. GCN Espacio-temporales (ST-GCN) : es el trabajo pionero para el reconocimiento de acciones basado en el esqueleto humano. En este método, el esqueleto del cuerpo humano se trata

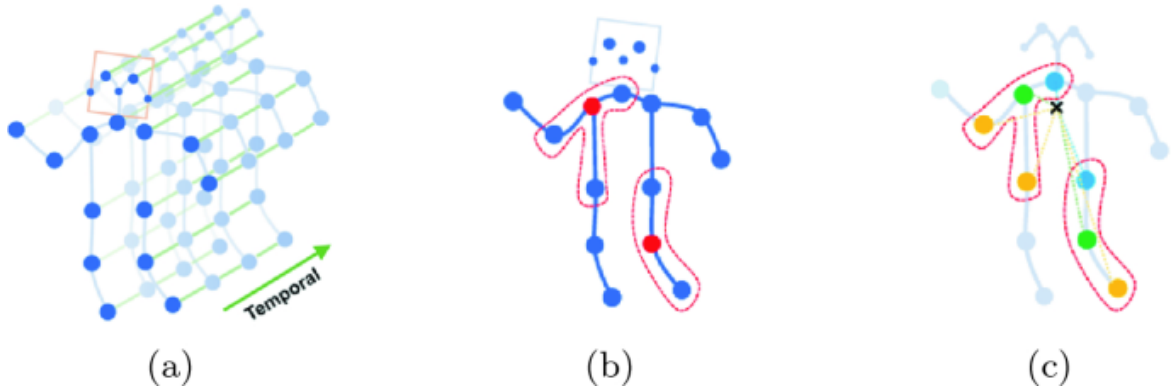


Figura 3.2: Ilustración de la superposición espacio-temporal de fotogramas de esqueleto Yan et al. (2018). (a) Gráfico espacio-temporal para una secuencia de esqueleto. (b) Particionamiento por distancia, el nodo raíz está a distancia 0, mientras que los nodos vecinos están a distancia 1. (c) Particionamiento espacial, los nodos están etiquetados según sus distancias al centro de gravedad del esqueleto..

como un grafo fijo hasta N fotogramas. La relación espacial entre las articulaciones se establece mediante aristas espaciales dentro del fotograma, mientras que la relación temporal entre las articulaciones se establece mediante aristas temporales entre los fotogramas.

2. GCN de dos multi-flujos (*two-multistream*): Intenta modelar información complementaria (de las articulaciones y los huesos) del esqueleto humano para el reconocimiento de acciones. Las arquitecturas de dos flujos o múltiples flujos constan de flujos separados de convoluciones de grafos, que luego se concatenan y se utiliza la función softmax para la clasificación.

3. GCN de atención recurrente: incluye la red recurrente o la red de atención en la arquitectura. Las redes de atención se utilizan para enfatizar las partes más relevantes de un esqueleto humano, lo cual se implementa mayormente utilizando arquitecturas recurrentes o LSTM.

4. *Encoder-decoder* GCN: Son métodos de aprendizaje no supervisado para codificar nodos/grafos en un espacio vectorial latente (aprendiendo una incrustación de red) y luego se reconstruyen los grafos a partir de la información codificada.

5. Otras soluciones GCN: Aquellas que no entran en ninguna de las anteriores categorías.

Las dos primeras se describen con más detalle a en la siguientes subsecciones, aunque antes se va a dar detalle sobre la terminología a usar.

El grafo G viene representado por el par $G = (V, E)$ donde V son los vértices o nodos del grafo y E son las aristas que los unen y para cada instante t se va a denominar $G_t = (V, E)_t$. Una arista $e_{ij} = (v_i v_j)$ es aquella que une el nodo v_i con el v_j. También se define la matriz de adyacencias A como:

$$A_{ij} = \begin{cases} 1 & \text{si } e_{ij} \in S \\ 0 & \text{si } e_{ij} \notin S. \end{cases} \tag{3.1}$$

Con todo esto tenemos una red convolucional de grafos donde las articulaciones son los vértices y los huesos las aristas, con matriz de adyacencia $A \in \mathbb{R}^{V \times V \times k}$ la matriz de pesos convolucionales aprendibles $A \in \mathbb{R}^{C_i \times C_o \times S}$ donde k son las diferentes particiones, S, C_i y C_o son el tamaño del núcleo, canales de entrada y canales de salida respectivamente. La ecuación general que define la operación de cada capa viene dada por:

$$Z = \sigma\left(\sum_{s=1}^{S} \hat{A} X W_s\right), \tag{3.2}$$

donde $\hat{A}$ es la matriz de adyacencia normalizada, $\sigma(\cdot)$ la función de activación, $X \in \mathbb{R}^{k \times V \times T \times C_i}$ son las características de entrada donde T es el fotograma y $Z \in \mathbb{R}^{V \times T \times C_o}$ es la matriz de características de salida.

GCN Espacio-temporales (ST-GCN) Las ST-GCN toman el esqueleto corporal de entrada como un grafo espacio-temporal con aristas espaciales y temporales entre los puntos del esqueleto. Un grafo espacio-temporal se representa como $\zeta = \{G_1, G_2, \ldots, G_n\}$, donde $G_1, G_2, \ldots, G_n$ es el grafo de esqueleto de entrada en cada fotograma y N es el número total de fotogramas. Un diagrama esquemático para este método se muestra en la figura 3.3(a). La principal ventaja de ST-GCN es que tiene un diseño genérico para aprender automáticamente patrones espaciales y temporales de los datos de entrada. En segundo lugar, el GCN espacio-temporal tiene una mayor capacidad expresiva y de generalización; además, funciona de manera complementaria, análoga a la información RGB y de flujo óptico. El trabajo pionero que utiliza ST-GCN fue propuesto por (Yan et al., 2018). Los autores utilizaron un esqueleto de video de entrada con un grafo espacio-temporal con muchos nodos y aristas. Las aristas dentro de un fotograma contribuyen a los detalles espaciales, mientras que las aristas entre fotogramas proporcionan la información temporal. Esta representación espacio-temporal es intuitiva para modelar el reconocimiento de acciones basado en esqueletos, como se muestra en la figura 3.4. El siguiente modelo matemático fue creado para esta arquitectura espacio-temporal:

$$Y_{out} = \sum_{j} D^{-1/2} A_j D^{-1/2} X_{in} W_j^l. \tag{3.3}$$

Donde Y_{out} es la salida de la red, A es la matriz de adyacencia, D es el grado de la matriz, X_{in} son las características de entrada de la capa X y W^l la matriz de pesos en la capa X.

ST-GCN toma los conceptos de la convolución 2D para la implementación de la función de muestreo y la función de pesos. El mapa de características en CNN suele ser de tamaño reducido, pero para las GCN, un relleno (*padding*) adecuado y establecer el paso (*stride*) como 1 resulta en el mapa de características de salida del mismo tamaño que el del mapa de características de entrada.

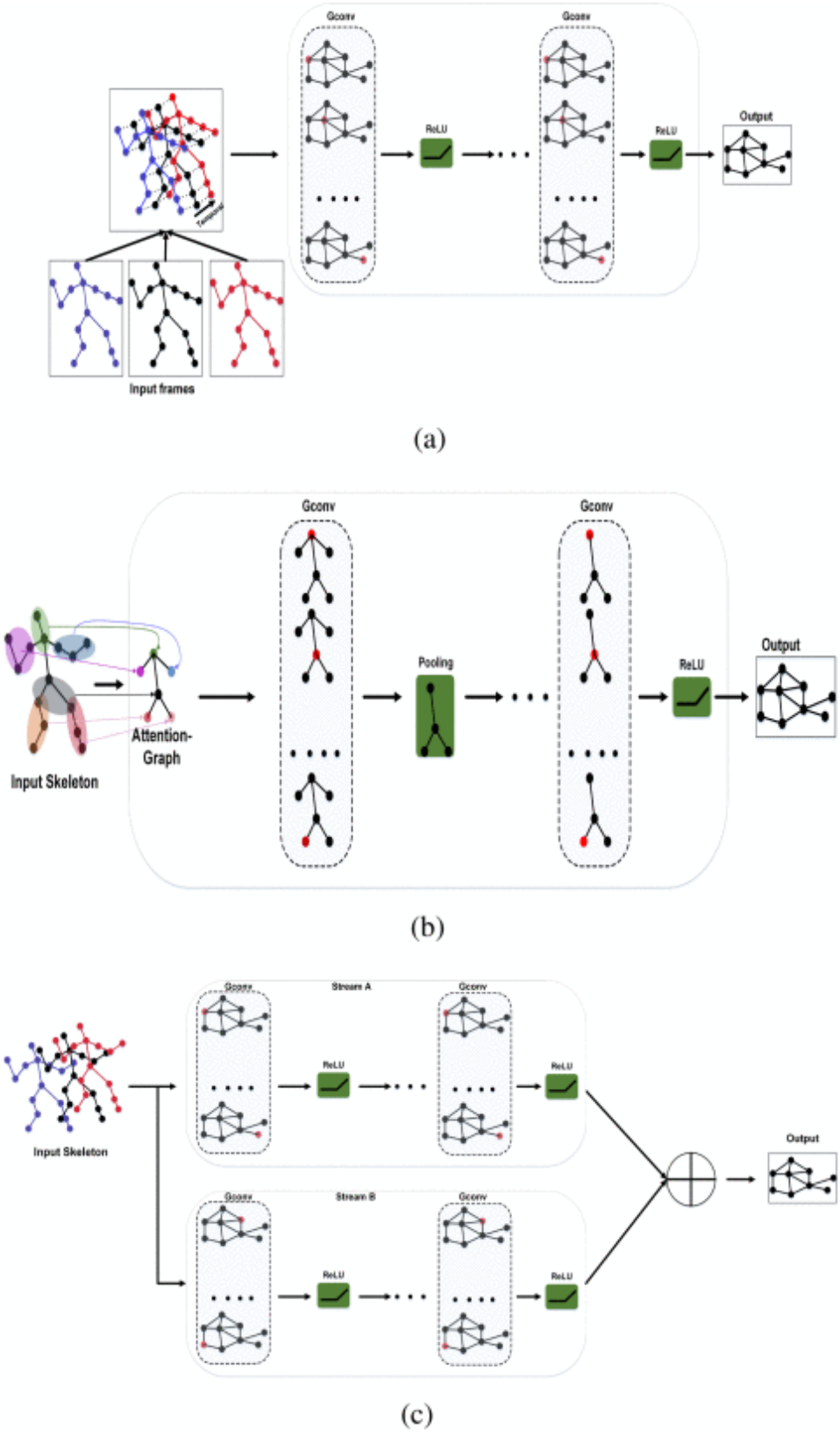


Figura 3.3: Diagrama esquemático ilustrando diferentes categorías de HAR mediante GCN. (a) Esquema para GCN espacio-temporal (ST-GCN). (b) Esquema para GCN recurrente-atención. (c) Esquema para GCM de doble flujo de entrada.

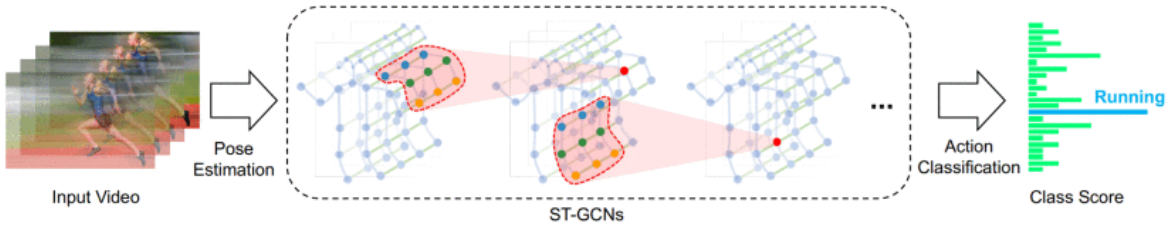


Figura 3.4: Ilustración de GCN espacio-temporal (Yan et al., 2018). Se aplican múltiples capas de ST-GCN y generan gradualmente mapas de características o *feature maps* de más alto nivel. Finalmente, un clasificador softmax estándar es utilizado para clasificar la acción en su categoría correspondiente.

(Chang and Liu, 2023) extendieron esta idea de ST-GCN para el reconocimiento de acciones basado en esqueletos al combinar los beneficios del filtrado convolucional local y la capacidad de aprendizaje de secuencias de media móvil autorregresiva. Esta red GCN consiste en campos receptivos locales y realiza recursivamente la convolución de grafos en los dominios espacial y temporal. En el reconocimiento de acciones basado en esqueletos, las articulaciones distantes contienen información importante para el reconocimiento de acciones; por ejemplo, en la acción de aplaudir, hay una relación importante entre las articulaciones distantes de las dos manos. (Li et al., 2019) abordaron este problema al diseñar una metodología de enrutamiento de gráficos espacio-temporales (STGR), que puede aprender de manera adaptativa relaciones de mayor orden entre las articulaciones distantes del esqueleto. STGR consta de enrutamiento gráfico espacial (SGR) y enrutamiento gráfico temporal (TGR). SGR incluye grupos de subgrupos para explotar la relación de conectividad entre las articulaciones distantes y TGR enfatiza la información estructural al correlacionar las trayectorias articulares temporales. STGR proporciona un campo receptivo bien equilibrado para diferentes articulaciones y es eficaz para el reconocimiento de acciones basado en esqueletos, definido como

$$\{g^{\text{spat}}, g^{\text{temp}}\} = f_{\text{STGR}}(X_{\text{in}}; \theta^{\text{spat}}, \theta^{\text{temp}}) \tag{6}$$

donde g^{spat} y g^{temp} son las topologías de gráficos espaciales y temporales, respectivamente y θ^{spat} y $\theta^{\text{temp}})$ representan los parámetros correspondientes. g^{spat} y g^{temp} se concatenan con el g^{default} para formar un conjunto de grafos $S = \{g^{\text{default}}, g^{\text{spat}}, g^{\text{temp}}\}$. Este concepto de STGR se explica claramente en la figura 3.5.

GCN de doble flujo (*two-stream*) El modelo *de GCN multistream* incluye dos o más flujos de redes de convolución de grafos para el reconocimiento de acciones. Cada flujo se activa con diferentes entradas (por ejemplo, articulaciones o huesos) y luego se enfatizan las características en cada flujo. Un diagrama esquemático para este método se muestra en la figura 3.3(c). Las técnicas *multistream* son métodos basados en datos y que con topologías de aprendizaje, lo que brinda una estructura de gráficos flexible. La otra ventaja principal de esta categorización es que podría involucrar información de segundo orden, como la longitud de los huesos y direcciones.

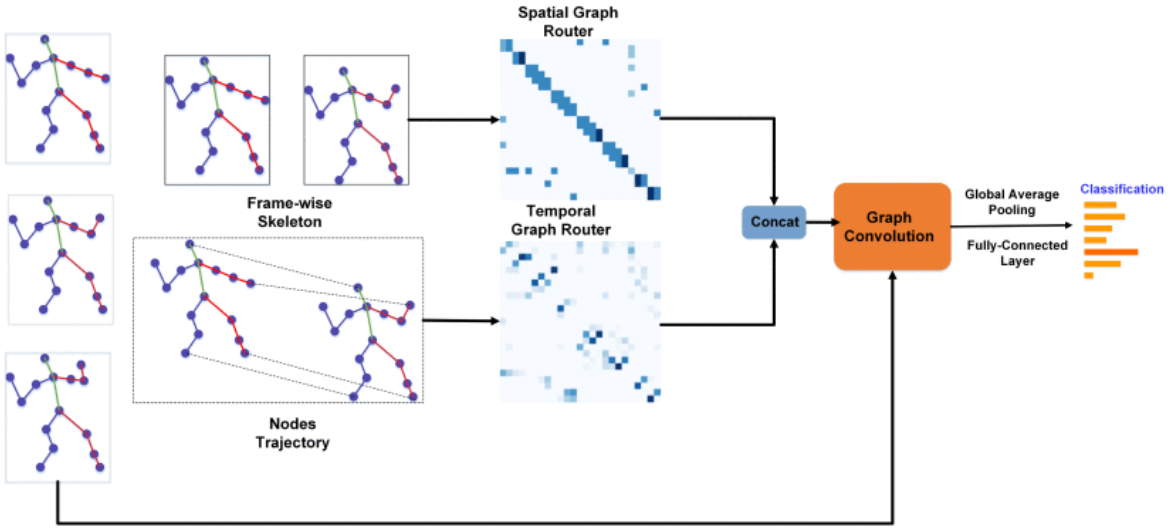


Figura 3.5: Ilustración de STGR (Li et al., 2019). SGR y TGR producen nuevos grafos de conectividad de articulaciones de esqueleto, respectivamente. ST-GCN recibe estos grafos y genera la clase de la acción.

Utilizando información de primer y segundo orden de las articulaciones y los huesos, (Shi et al., 2019b) propusieron un nuevo modelo de grafos de convolución adaptable de dos flujos (2s-AGCN) para el reconocimiento de acciones basado en esqueletos. Los autores utilizaron un flujo de huesos y un flujo de articulaciones como marco de dos flujos, que es efectivo para modelar simultáneamente la información de primer y segundo orden y promete un considerable aumento en la precisión del reconocimiento de acciones. En este método, la topología del grafo se optimiza junto con otros parámetros de la red de manera de aprendizaje de extremo a extremo. El 2s-AGCN se define matemáticamente como:

$$H^{l+1} = \sum_{k}^{K_v} W_k^l H^l (A_k + B_k + C_k). \tag{12}$$

Aquí, la matriz de adyacencia incluye las siguientes tres matrices: A_k, B_k y C_k. A_k es la matriz de adyacencia original $N \times N$. B_k también es una matriz de adyacencia $N \times N$, aprendida durante el proceso de entrenamiento. C_k es una matriz de similitud de nodos $N \times N$. La arquitectura de dos flujos es un trabajo pionero que proporcionó una base sólida para el reconocimiento de acciones basado en GCN utilizando el método de dos y múltiples flujos (Li et al., 2020a; Luo et al., 2019; Shi et al., 2019a; Gao et al., 2019). Esta arquitectura de dos flujos se muestra en la figura . En contraste con las estructuras de gráficos fijos de ST-GCN en (Yan et al., 2018), 2s-AGCN implica estructuras de grafos óptimas basadas en datos para diferentes clases de acciones. Otro modelo de dos flujos fue propuesto como grafos de nodos y aristas por (Li et al., 2020a). Este método incorpora las siguientes tres estrategias de partición:

1. Etiquetado único

2. Particionamiento por distancia

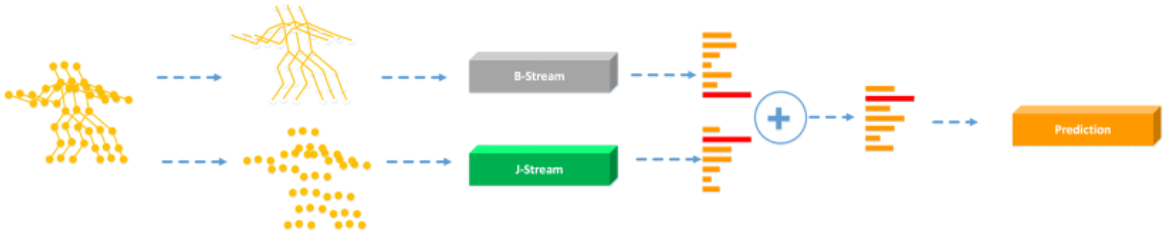


Figura 3.6: Arquitectura general de 2s-AGCN (Shi et al., 2019b). Las puntuaciones de los dos flujos son fusionadas para obtener una predicción final.

3. Particionamiento de configuración espacial.

En contraste con la arquitectura de dos flujos propuesta en (Shi et al., 2019b), la representación de aristas del grafo se realiza formulando los centros de los huesos. El flujo de nodos y el flujo de aristas se fusionan, luego se utilizan funciones de agrupación (*pooling*) y *softmax* para obtener la probabilidad de predicción para cada clase. (Luo et al., 2019) propusieron un novedoso grafo de convolución basado en razonamiento de interacciones. Este marco de dos flujos puede modelar la interacción entre objetos y parches de escena. Un flujo de CNN extrae características de objetos, mientras que un detector de escena discriminatorio modela los parches de escena. Luego, la interacción entre objetos y escenas se modela como un grafo, que se alimenta a GCN para la predicción de clases. (Shi et al., 2019a) representaron los datos de esqueleto como un grafo acíclico dirigido, determinando la dependencia cinemática entre las articulaciones, los huesos y su relación. La estructura topológica de un grafo es adaptable durante el entrenamiento, lo que mejora significativamente el rendimiento. Esta técnica funciona de manera de dos flujos, que incluye el movimiento de las articulaciones y las deformaciones de los huesos para el reconocimiento de acciones. La salida de la red de dos flujos se fusiona utilizando una capa softmax para la clasificación final.

Un estudio que utiliza el aprendizaje *zero-shot* con GCN de dos flujos fue realizado por (Gao et al., 2019). En este artículo, se formulan grafos de conocimiento para modelar relaciones explícitas utilizando el reconocimiento de acciones *zero-shot*. Tales gráficos de conocimiento estructurado pueden modelar relaciones entre acciones-atributos, acciones-acciones y atributos-atributos. El método propuesto funciona como un GCN de dos flujos, con una rama de clasificación y una rama de instancias. Los autores validaron los resultados de este método en *datasets* de referencia. La rama clasificadora toma información de sujetos y escenas como vectores embebidos semánticos para el reconocimiento de acciones, mientras que la rama de instancias modela el embebido (*embedding*) de atributos. Este método propuesto funciona de manera de extremo a extremo de modo que los clasificadores generados y las características de las instancias se adapten y cooperen para la clasificación final. (Gao et al., 2020) extendieron la idea de (Gao et al., 2019) utilizando *embeddings* de palabras y aprendizaje *zero-shot*. En este artículo, se introduce un método de paso de mensajes impulsado por la tarea, denominado GCN de prototipo-muestra, con una rama de prototipo y una rama de muestra. La rama de prototipo se encarga de aprender la representación de las categorías de

video, mientras que la rama de muestra genera muestras de video aprovechando la semántica de los objetos. Los autores evaluaron su método en cinco conjuntos de datos de referencia y demostraron que el marco propuesto puede funcionar mejor que los métodos contemporáneos.

Otras arquitecturas involucran el uso de características de orden superior, como velocidades y aceleraciones de articulaciones y huesos (Dong et al., 2020), para mejorar la robustez del modelo. También se han explorado diferentes modalidades de entrada, como coordenadas articulares, direcciones de huesos y desplazamientos temporales de articulaciones, y se han utilizado múltiples escalas y flujos para capturar información enriquecida y multinivel. Además, se han investigado métodos para enriquecer la información topológica utilizando la centralidad de nodos, huesos y subgrafos en el dominio espacial. Estas arquitecturas han demostrado un rendimiento prometedor en conjuntos de datos de reconocimiento de acciones de referencia y han mejorado el reconocimiento de acciones basado en el esqueleto al capturar mejor las relaciones espaciales y temporales entre las articulaciones del cuerpo.

A principios de este año (Liu et al., 2023) propusieron un modelo que surge como un incremento y mejora de (Yan et al., 2018) permitiendo mayor complejidad gracias que utiliza técnicas de entrenamiento más eficientes y será explicado con más detalle más adelante.

3.3.2. Dataset comunes

Entre los datasets más relevantes en este campo de reconocimiento de acciones a partir de GCN se encuentran los siguientes:

- NTU RGB+D (Shahroudy et al., 2016): Es un conjunto de datos multimodal que contiene videos de acciones realizadas por 40 sujetos en diferentes ángulos y utilizando cámaras Kinect v.2. Se utilizan evaluaciones cruzadas de sujeto y vista, con 60 clases de acciones diferentes.

- Kinetics-Skeleton (Kay et al., 2017): Es un conjunto de datos con videos de acciones humanas de YouTube. Se utiliza la información de esqueleto obtenida mediante OpenPose para representar las articulaciones de las personas en el video. Contiene 400 clases de acciones y se reportan las precisión Top-1 y Top-5.

- NTU-RGBD-120 (Liu et al., 2019): Una extensión del conjunto de datos NTU RGB+D-60 con 120 clases de acciones diferentes y 114,480 videos. Contiene datos de RGB, secuencias de profundidad, datos de esqueleto e imágenes infrarrojas.

- SYSU-3D (Hu et al., 2015): Un conjunto de datos con 480 videos de 40 sujetos realizando 12 actividades diferentes. Contiene datos de esqueleto con 30 coordenadas de 20 articulaciones diferentes por cuadro.

- Northwestern-UCLA (Wang et al., 2014): Contiene 1494 videos de diez acciones diferentes, capturados simultáneamente por tres cámaras Kinect. Cada secuencia de video contiene datos

de RGB y esqueleto.

- UT-Kinect (Hussein et al., 2013): Un conjunto de datos de esqueleto con 200 secuencias de video que representan diez clases de acciones realizadas por diez sujetos diferentes.
- MSR Action3D (Li et al., 2010): Contiene 557 videos de interacciones de dos personas con 20 clases de acciones diferentes realizadas por diez sujetos.

Capítulo 4

Propuesta y metodología

En este capítulo se describirán con detalle los dos modelos utilizados y las técnicas empleadas para generar una respuesta unificada entre los distintos sensores. Como se mencionó en el capítulo 2.3, los dispositivos utilizados son los sensores IMU atr01 y atr02 y los *keypoints* obtenidos con el sensor kinect.

4.1. Modelo sensores IMU

Debido a su buen rendimiento, eficiencia computacional y comportamiento en paradigmas multimodales se ha decidido utilizar Deep ConvLSTM (Ordóñez and Roggen, 2016). Los autores introducen un nuevo marco de trabajo de DNN (Redes Neuronales Profundas) para el reconocimiento de actividades en dispositivos *wearables*, al que han denominado DeepConvLSTM. Esta arquitectura combina capas convolucionales y recurrentes. Las capas convolucionales actúan como extractores de características y proporcionan representaciones abstractas de los datos del sensor de entrada en forma de mapas de características. Las capas recurrentes modelan la dinámica temporal de la activación de los mapas de características. En este marco, las capas convolucionales no incluyen una operación de agrupación. La estructura de red se muestra en la figura 4.1, con cuatro capas convolucionales y tres capas densas. La entrada se procesa de manera secuencial por capas, donde cada capa proporciona la representación de la entrada que se utilizará como datos para la siguiente capa. El número de núcleos en las capas convolucionales y las unidades de procesamiento en las capas densas es el mismo para ambos casos. La principal diferencia entre DeepConvLSTM y un modelo CNN tradicional es la topología de las capas densas. En el caso de DeepConvLSTM, las unidades de estas capas son celdas recurrentes LSTM, y en el caso del modelo CNN, las unidades no son recurrentes y están completamente conectadas.

La entrada a la red consiste en una secuencia de datos. La secuencia es una serie temporal corta extraída de los datos del sensor utilizando un enfoque de ventana deslizante compuesta por varios

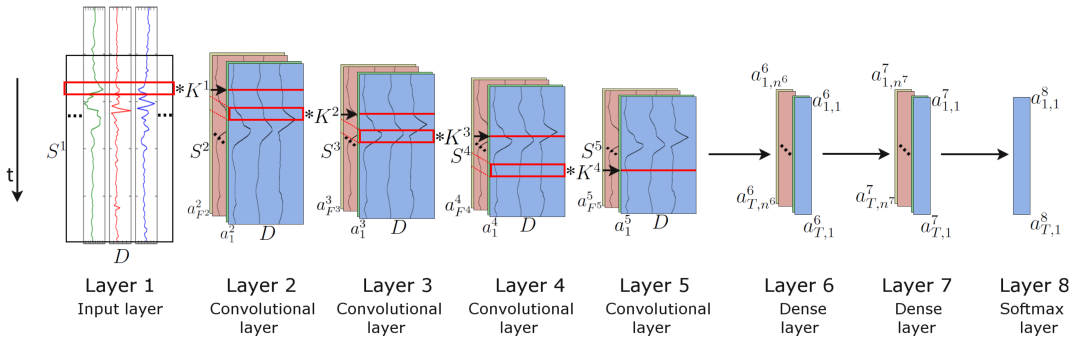


Figura 4.1: Arquitectura del marco de trabajo DeepConvLSTM (Conv, convolucional) para el reconocimiento de actividades. Desde la izquierda, las señales provenientes de los sensores son procesadas por cuatro capas convolucionales, que permiten aprender características de los datos. Luego, dos capas densas realizan una transformación no lineal, lo que da como resultado el resultado de clasificación con una capa de salida de regresión logística softmax en el lado derecho. La entrada en la Capa 1 corresponde a datos de sensor de tamaño $D \times S^1$ donde D denota el número de canales de sensor y S^l la longitud de los mapas de características en la capa l. Las Capas 2-5 son capas convolucionales. K^l denota los núcleos en la capa l (representados como cuadrados rojos). F^l denota el número de mapas de características en la capa l. En las capas convolucionales, a_i^l denota la activación que define el mapa de características i en la capa l. Las Capas 6 y 7 son capas densas. En las capas densas, $a_{t,i}^l$ denota la activación de la unidad i en la capa oculta l en el tiempo t. El eje de tiempo es vertical.

canales de sensor. El número de canales de sensor se denota como D. Dentro de esa secuencia, todos los canales tienen el mismo número de muestras S^1. La longitud de los mapas de características S^l varía en diferentes capas convolucionales. La convolución solo se calcula donde la entrada y el núcleo se superponen completamente. Por lo tanto, la longitud de un mapa de características está definida por:

$$S^{(l+1)} = S^l - P^l + 1 \tag{4.1}$$

donde P^l es la longitud de los núcleos en la capa l. Dicha longitud es la misma para todas las capas convolucionales, definidas como $P^l = 5, \forall l = 2, ..., 5$.

DeepConvLSTM es una red neuronal profunda (DNN) que comprende capas convolucionales, recurrentes y de softmax. En primer lugar, los datos del sensor se transforman a través de cuatro operaciones convolucionales. Las capas convolucionales procesan la entrada solo a lo largo del eje que representa el tiempo. El número de canales de sensor es el mismo para cada mapa de características en todas las capas. En la figura 4.1, los operadores de convolución se muestran como '*', que se aplican a un núcleo cuyo tamaño está delineado por los rectángulos rojos.

Estas capas convolucionales emplean unidades lineales rectificadas (ReLUs) para calcular los mapas de características, cuya función no lineal se define como $\sigma(x) = max(0, x)$. Las capas 6 y 7 son capas densas recurrentes. La elección del número de capas recurrentes se realiza siguiendo los resultados presentados por (Karpathy et al., 2015), donde los autores mostraron que una profundidad

de al menos dos capas recurrentes es beneficiosa al procesar datos secuenciales.

Las capas densas recurrentes adaptan su estado interno después de cada paso de tiempo. Aquí, las entradas de la Capa 6 en el tiempo t son los elementos de todos los mapas de características en la Capa 5 en el tiempo t, con $t = 1...T$ y $T = S^5$. La activación de las unidades recurrentes se calcula utilizando la función tangente hiperbólica. La salida del modelo se obtiene de una capa *softmax* (una capa densa con una función de activación *softmax*), lo que produce una distribución de probabilidad de clases para cada paso de tiempo individual t. La descripción resumida de la arquitectura del modelo sería: $C(64) - C(64) - C(64) - C(64) - R(128) - R(128) - Sm$ donde $C(F^l)$ representa la capa convolucional l con F^l mapas de características, $R(n^l)$ la capa recurrente LSTM con n^l células y Sm el clasificador *softmax*.

4.2. Modelo sobre *keypoints*

Como modelo en la parte de visión con *keypoints* dados se ha escogido el trabajo de (Liu et al., 2023). Como información preliminar, los formatos de datos son la normalización del preprocesamiento del esqueleto 3D en (Chen et al., 2021) y el esqueleto 2D generado por HRNet. Unificamos la forma de la secuencia del esqueleto como $\chi \in \mathbb{R}^{3\times M\times T\times V}$, donde M, T y V denotan el número de personas, fotogramas y articulaciones, respectivamente. Suponemos que $\chi[:, m_i, t_i, v_i]$ es la coordenada de la articulación v_i de la persona m_i en el fotograma t_i. Para el esqueleto 3D, $\chi[:, m_i, t_i, v_i] \in \{x, y, z\}$ representa las coordenadas 3D. Para el esqueleto 2D, $\chi[:, m_i, t_i, v_i] \in \{x, y, c\}$ representa las coordenadas 2D y la confianza de la articulación. Luego, formateamos χ en 4 características de entrada: articulación χ_j, hueso χ_b, movimiento de articulación χ_{jm} y movimiento de hueso χ_{bm}.

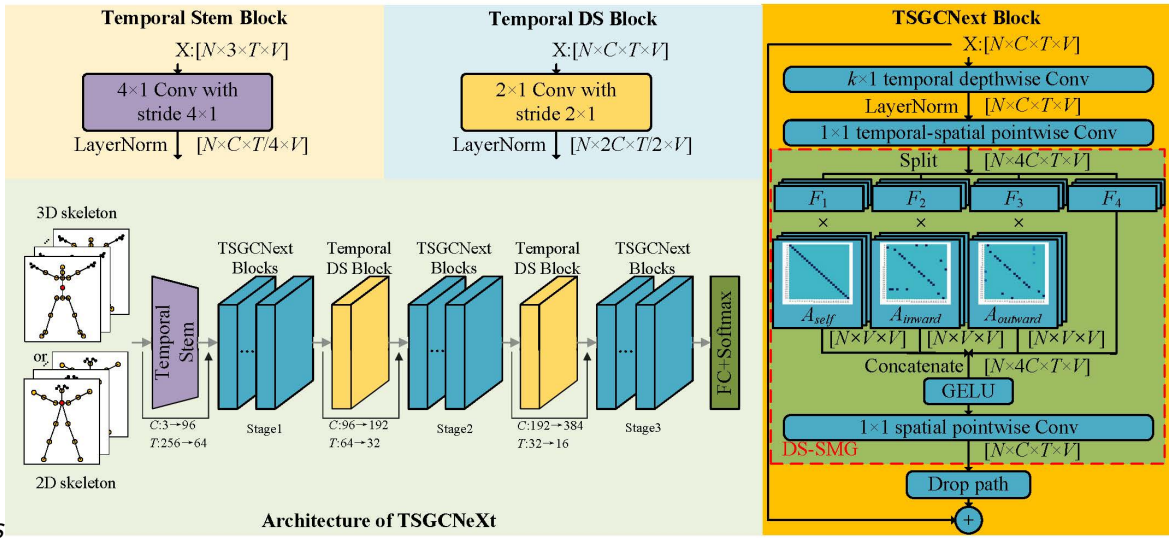
La arquitectura de TSGCNeXt puede verse en la figura 4.2, donde está la Convolución Dinámica-Estática con Multi-Grafo Separado (DS-SMG), que es la pieza clave. También se cuenta con una estrategia de convoluciones de grafo que permite reducir los costes de entrenamiento. Por último, se presenta la reconstrucción de la arquitectura general con tres bloques de aprendizaje espacio-temporal para lograr un aprendizaje eficiente en series temporales largas.

4.2.1. DS-SMG

Aquí tienes las ecuaciones en formato LaTeX:

La matriz de adyacencia Multi-Grafo es una de las técnicas más importantes utilizadas por la convolución de gráficos esqueléticos para aprender las relaciones entre los nodos. En este caso A se denota como:

$$A = \{A_s, A_i, A_o\} \tag{4.2}$$



keypoints

Figura 4.2: El flujo general de TSGCNeXt, que incluye la arquitectura y los detalles de los bloques. El DS-SMG en el cuadro de puntos rojos es la Convolución Dinámica-Estática con Multi-Grafo Separado. La relación de cálculo de las tres etapas está diseñada para ser 2:5:2 o 4:3:2.

donde A_s, A_i, A_o corresponden a las relaciones de autoreferencia, internas y externas respectivamente. Por lo tanto, en la arquitectura clásica de convolución de gráficos, la ecuación de 3.3.1.1 se expresa como:

$$Z = \sigma\left(\hat{A}_s XW + \hat{A}_i XW + \hat{A}_o XW\right) \tag{4.3}$$

que agrega matrices de adyacencia múltiples con multicolinealidad. Permite una relación lineal entre diferentes topologías en lugar de topologías no compartidas. En CTR-GCN, las topologías no compartidas pueden mejorar el aprendizaje de gráficos. Así que se diseñó la Convolución Dinámica-Estática con Multi-Grafo Separado para explorar la agregación de información independiente de matrices de adyacencia múltiples. El mecanismo es el siguiente:

$$Z = \sigma\left(\hat{A}_s\sigma(F_1), \hat{A}_i\sigma(F_2), \hat{A}_o\sigma(F_3), \sigma(F_4)\right) W_\gamma \tag{4.4}$$

donde $W^j \in \mathbb{R}^{Ci\times Ci}$ corresponde a $W[:, (j-1)Ci : jCi]$, $F_i = \sigma(XW_i)$, y $W \in \mathbb{R}^{4Ci\times Ci}$ corresponde a la convolución punto a punto espacial. La entrada X se asigna primero mediante una matriz de adyacencia múltiple para generar características separadas para cada adyacencia. Estas características independientes se agregan luego mediante una convolución punto a punto espacial para generar un nuevo conjunto de puntos Z. Cabe señalar que las características se dividen por igual en cuatro copias. Las primeras tres copias se utilizan para aprender las relaciones topológicas de los nodos individuales bajo topologías inicializadas diferentes, mientras que la última copia F_4 no tiene matriz. Esto permite que Z siempre contenga toda la información de los nodos de X incluso aunque la matriz de adyacencia filtre la información de estos.

4.2.2. Mecanismo de aceleración del entrenamiento

Una de las características más destacables de las GCN respecto a las CNN es la multiplicación matricial entre la matriz de adyacencias y las matrices de características. Este proceso se manifiesta mediante la operación *einsum* (Convenio de suma de Einstein) donde la salida Y de una convolución de grafo con k subgrafos puede expresarse como:

$$Y = enisum(\text{“}VUk, NCTVk\text{ß}NCTUk\text{”}, A, X). \tag{4.5}$$

El tiempo de entrenamiento se concentra principalmente en el cálculo inverso de la operación, por esta razón, se diseñó una operación equivalente más eficiente para el cálculo propagación hacia atrás durante el entrenamiento.

$$\tilde{A} = repeat(A) = \{\tilde{A}_1, \tilde{A}_2, ..., \tilde{A}_N | \tilde{A}_i = A\}, \tag{4.6}$$

$$\tilde{Y} = enisum(\text{“}NVUk, NCTVk\text{ß}NCTUk\text{”}, \tilde{A}, X). \tag{4.7}$$

Repitiendo A N veces se logra agilizar a la hora de realizar las derivadas que intervienen en la backpropagation pasando de tener que calcular NC derivadas a únicamente C y generando resultados equivalentes sencillamente demostrables mediante la regla de la cadena.

4.2.3. Reconstrucción general de la arquitectura

La mayoría de las redes basadas en GCN para la clasificación de acciones a partir de la pose están diseñadas en base a STGCN. Los bloques se dividen en 3 etapas y el número de canales base es 64. Inspirado por ConvNeXt, el número de canales base se establece primero en 96 para aumentar la riqueza de las características de aprendizaje. Luego exploramos el efecto de las tasas de cálculo de etapa (SCR) con un número constante de capas. TSGCNeXt utiliza dos configuraciones principales: 2:5:2 y 4:3:2. 2:5:2 sigue el consenso de que un número grande de etapa 3 se considera más efectivo para las redes visuales con 4 etapas. 4:3:2 se ajusta al principio de las GCN anteriores. Ambas configuraciones se entrenan en diferentes conjuntos de datos para lograr mejores resultados.

A diferencia de trabajos anteriores que solo contienen bloques de convolución de grafo espacio-temporal y clasificaciones, TSGCNeXt introduce los principios de *Patchify Stem, Separate Down Sampling* y CovNeXtify block de ConvNeXt (Liu et al., 2022), diseñando el bloque *Temporal Stem*, el bloque *Separate Temporal Down Sampling* y el bloque TSGCNeXt respectivamente.

- Bloque *Temporal Stem*: La capa inicial se utiliza para transformar las características de entrada en características apropiadas mediante el muestreo descendente. Sin embargo, en una GCN de esqueleto típica, las características temporales suelen extraerse directamente sin usar *stem*, lo que limita la granularidad de la información temporal. Por lo tanto, el bloque *Temporal Stem*

expande la dimensión temporal de la entrada para aprender características temporales a largo plazo. Como se muestra en la figura 4.2, el bloque Temporal Stem consta de una capa Conv 4×1 con paso 4×1 y una capa LayerNorm. La longitud de secuencia T se establece en 256 para asegurar que las dimensiones de las características temporales correspondan a una complejidad computacional comparable con trabajos anteriores. Dado que la convolución en el bloque *Temporal Stem* no se superpone, el diseño "*patchify*" es más eficiente para construir características.

- Bloque de *Separate Down Sampling*: El muestreo descendente está estrechamente relacionado con las capas de convolución espacio-temporal en las GCN anteriores en la capa final de la etapa. Sin embargo, esto a menudo entra en conflicto con la estructura residual, ya que las características muestreadas no coinciden con las características de entrada. Por lo tanto, se utiliza un bloque de *Separate Down Sampling* en las estructuras de red modernas agregando un módulo de convolución muestreado entre etapa y etapa. Como se muestra en la figura 4.2, el tamaño del núcleo y el paso del bloque de *Separate Down Sampling* son ambos de 2×1, agregando características en la dimensión temporal.

- Bloque TSGCNeXt. La plantilla de diseño del módulo TSGCNeXt sigue el bloque ConvNextify con una convolución profunda temporal de gran profundidad y una convolución de punto espacial MLPfiy, como se muestra en la figura 4.2. Además, las capas de activación y normalización utilizan GLUE y LN para complementar la estrategia de entrenamiento distribuido. A diferencia de la sustitución directa anterior del nuevo módulo con módulos de convolución espacial y temporal separados, integramos orgánicamente el proceso de extracción de información espacio-temporal en un solo módulo ConvNextify. Los módulos anteriores de convolución de grafo espacio-temporal tienden a extraer características espaciales primero y luego características temporales. Sin embargo, extraemos las características temporales antes de extraer las características espaciales, de ahí el término de convolución temporal-espacial (TS). Específicamente, la primera capa es una convolución profunda temporal de $k \times 1$, donde k corresponde al campo receptivo temporal. Cada núcleo de convolución está conectado de forma independiente a un conjunto de canales de entrada y salida. Esto reduce el número de parámetros y cumple con el mecanismo de topología no compartida. Luego, una convolución punto a punto temporal-espacial de 1×1 establece relaciones entre canales mientras expande la dimensionalidad de las características. Después, aplicamos el mecanismo de Convolución Dinámica-Estática con Multi-Grafo Separado para extraer características espaciales y agregamos rutas de descarte para mejorar la capacidad de modelado no lineal del módulo.

4.3. Entrenamiento y unificación de modelos

Ambos modelos fueron entrenados por separado en PyTorch utilizando en gran medida el marco de trabajo y el conjunto de herramientas proporcionado por Openpack sobre una GPU Nvidia RTX A2000 de 4GB de RAM.

El modelo DeepConvLSTM ha sido entrenado con tamaño de lote de 32, con una ventana deslizante de 1800 muestras sin solapamiento, que al ser recogidas a 30Hz equivalen a muestras de 60 segundos. El entrenamiento se ha realizado por separado en dos partes tanto con los datos del giroscopio como con los del acelerómetro. De optimizador se ha usado Adam con decaimiento en las épocas 50 y 75 hasta un total de 100 épocas, donde se almacenaban los pesos del mínimo absoluto en la función de pérdida. Esta función de pérdida consistía en una combinación de entropía cruzada junto a una función de suavizado como proponen (Farha and Gall, 2019). La función de coste para la entropía cruzada es:

$$\mathcal{L}_{cls} = \frac{1}{T}\sum_{t} -\log(y_{t,c}), \tag{4.8}$$

donde $y_{t,c}$ es la probabilidad de predicción de la etiqueta real en el instante t. Esta función tiene un buen funcionamiento, pero su rendimiento es peor cuando se prodice sobre-fragmentación (el sistema divide la predicción en demasiadas partes asignando una clase errónea a algunas atrapadas en el interior de un fragmento de una única clase) y la función de coste de suavizado permite esquivar este problema aplicando un error cuadrático medio truncado de la siguiente manera:

$$\mathcal{L}_{T-MSE} = \frac{1}{TC}\sum_{t,c} -\tilde{\Delta}^2_{t,c}, \tilde{\Delta}^2_{t,c} = \begin{cases} \Delta^2_{t,c} & \text{si } \Delta^2_{t,c} \leq \tau \\ \tau & \text{si } Cualquer\ otro \end{cases}, \Delta^2_{t,c} = |\log y_{t,c} - \log y_{t-1,c}|, \tag{4.9}$$

Donde T es la duración de la secuencia, C es el número de clases e $y_{t,c}$ es la probabilidad de la clase c en el instante t. Se puede ver que esta función es muy similar a la función de coste de divergencia Kullback-Leibler.

El modelo TSGCNeXt ha sido entrenado con tamaño de lote de 128, con una ventana deslizante de 64 muestras con 5 de desplazamiento, que al ser recogidas a 15 fps equivalen a muestras de 8,53 segundos. De optimizador se ha usado Adam con decaimiento en las épocas 30, 50 y 80 hasta un total de 100 épocas, donde se almacenaban los pesos del mínimo absoluto en la función de pérdida. Como función de pérdida se ha utilizado entropía cruzada. Este entrenamiento se ha dividido en 4 partes, una para huesos, otra para articulaciones y otra dos para el movimiento de ambas como se mencionó previamente en el apartado 4.2.

Como métricas de evaluación se utilizarán principalmente precisión, exhaustividad (*recall*) y valor-F (*f1-score*). Esta última será la utilizada para comparar con el resto de participantes tanto a nivel *macro average* como ponderado.

La estructura de los ficheros está basada en la propuesta en el GitHub de TSGCNeXt[1], donde se ha utilizado en gran medida la funcionalidad ofrecida por el *toolkit* de openpack realizando modificaciones. Entre ellas, la más relevante es la constitución del grafo, se ha realizado de manera personalizada un grafo basado únicamente en 15 *keypoints* descartando así aquellos correspondientes a ambos pies, pues no son relevantes en nuestra casuística e inducen a errores mayores ya que no son visibles en la imagen. Se ha modificado alguna de las funciones para poder realizar un paso de ventana deslizante inferior al tamaño de ventana para permitir solapamiento de cara al entrenamiento de *keypoints*.

Los modelos se entrenan con cada variante por separado dando un total de 8 entrenamientos (2 para cada sensor IMU y 4 para *keypoints*) y, una vez entrenados, se genera una predicción sobre el conjunto de evaluación propuesto por openpack para cada uno y además una sobre el conjunto de entrega (*submission*) que será explicado en el siguiente apartado. La salida de las predicciones de evaluación se almacena en un fichero csv indicando las métricas descritas anteriormente desglosando por operación y por sesión además de unas a nivel general. Las salidas de las predicciones sobre el conjunto de entrega se almacenan en formato *numpy* para un posterior ensamblaje conjunto que produce una salida unificada.

El ensamblaje se realiza una vez se han obtenido todos los resultados parciales. Primeramente, se han de equiparar todos los datos a un mismo eje temporal, pues las frecuentas de muestreo de los distintos sensores son diferentes. El *tooklit* de openpack ofrece herramientas para realizar este re-muestreo para hacer coincidir los datos provenientes de sensores IMU con los de *keypoints* y, a su vez, equipararlos con los datos de las etiquetas, que vienen con 1Hz. Finalmente, se aplica una suma ponderada y parametrizada con ponderaciones ajustables que son de 0,8 para los *keypoints* y 0,4 para los sensores IMU y generando así un fichero entregable listo para ser subido a la web de participación.

Para más detalles consultar el código que se adjunta con la memoria.

[1] https://github.com/vvhj/TSGCNeXt

Capítulo 5

Resultados y discusión

Tras los experimentos realizados en el capítulo 4 se muestran en este apartado los resultados generales obtenidos para cada componente y el resultado conjunto en los entrenamientos realizados habiendo pasado unas fases previas de ajustes de hiperparámetros. El formato vendrá dado por la salida de la función precision_recall_fscore_support de scikit-learn. Para los resultados sobre cada sesión acudir al apéndice donde se desglosarán en mayor detalle.

5.1. Resultados IMU

Para los sensores IMU tenemos las tablas 5.1, 5.2, 5.3 y 5.4.

Se puede apreciar que en general hay un equilibrio en precisión y *recall*, arrojando un valor-f similar a ambos. Se ve también que indistintamente del brazo medido se alcanzan métricas similares y que en todos los casos la operación peor clasificada es aquella en la que el operario coloca la etiqueta. Esto es comprensible, pues no requiere un movimiento de los brazos característico ya que tiene gran variabilidad dependiendo del tamaño del paquete, posición en la que se recoloca, etc. Aun así, estos resultados indican que se puede conocer qué operación se está realizando con una precisión mayor al 93 % (precisión) y que las acciones realizadas han sido correctamente clasificadas aproximadamente un 93 % de las ocasiones (exhaustividad).

5.2. Resultados keypoints

Para los keypoints tenemos las tablas 5.5, 5.6, 5.7 y 5.8:

Se puede comprobar que, al igual que en el caso de los resultados sobre sensores IMU, existe un equilibrio en precisión y *recall*, arrojando un valor-f similar a ambos. De la misma forma, la operación peor clasificada es aquella en la que el operario coloca la etiqueta seguida de la colocación de elementos en la caja. La explicación radica en que al realizar estas actividades las extremidades sufren oclusiones, algo que impide controlar la correcta posición y por tanto realizar la clasificación.

name	id	precision	recall	f1	support	key
avg/macro	-1	0,935	0,924	0,930		all
avg/weighted	-1	0,941	0,928	0,935		all
Picking	100	0,936	0,949	0,943	589	all
Relocate Item Label	200	0,963	0,917	0,940	881	all
Assemble Box	300	0,945	0,954	0,949	1023	all
Insert Items	400	0,931	0,908	0,919	489	all
Close Box	500	0,951	0,911	0,930	696	all
Attach Box Label	600	0,891	0,897	0,894	291	all
Scan Label	700	0,962	0,934	0,948	788	all
Attach Shipping Label	800	0,923	0,935	0,929	447	all
Put on Back Table	900	0,941	0,908	0,924	262	all
Fill out Order	1000	0,910	0,930	0,920	499	all

Tabla 5.1: Resultados sobre acelerómetro del sensor IMU de muñeca izquierda

name	id	precision	recall	f1	support	key
avg/macro	-1	0,935	0,925	0,930		all
avg/weighted	-1	0,940	0,929	0,935		all
Picking	100	0,924	0,949	0,936	589	all
Relocate Item Label	200	0,966	0,901	0,932	881	all
Assemble Box	300	0,944	0,941	0,943	1023	all
Insert Items	400	0,882	0,937	0,909	489	all
Close Box	500	0,962	0,944	0,953	696	all
Attach Box Label	600	0,904	0,876	0,890	291	all
Scan Label	700	0,957	0,943	0,950	788	all
Attach Shipping Label	800	0,938	0,949	0,943	447	all
Put on Back Table	900	0,937	0,901	0,918	262	all
Fill out Order	1000	0,932	0,912	0,922	499	all

Tabla 5.2: Resultados sobre giroscopio del sensor IMU de muñeca izquierda

name	id	precision	recall	f1	support	key
avg/macro	-1	0,937	0,928	0,932		all
avg/weighted	-1	0,945	0,932	0,938		all
Picking	100	0,935	0,954	0,945	589	all
Relocate Item Label	200	0,964	0,910	0,936	881	all
Assemble Box	300	0,959	0,949	0,954	1023	all
Insert Items	400	0,907	0,935	0,920	489	all
Close Box	500	0,964	0,925	0,944	696	all
Attach Box Label	600	0,889	0,883	0,886	291	all
Scan Label	700	0,960	0,952	0,956	788	all
Attach Shipping Label	800	0,927	0,931	0,929	447	all
Put on Back Table	900	0,927	0,916	0,921	262	all
Fill out Order	1000	0,933	0,924	0,928	499	all

Tabla 5.3: Resultados sobre acelerómetro del sensor IMU de muñeca derecha

name	id	precision	recall	f1	support	key
avg/macro	-1	0,930	0,925	0,927		all
avg/weighted	-1	0,936	0,925	0,930		all
Picking	100	0,924	0,951	0,937	589	all
Relocate Item Label	200	0,964	0,880	0,920	881	all
Assemble Box	300	0,925	0,934	0,929	1023	all
Insert Items	400	0,868	0,926	0,896	489	all
Close Box	500	0,956	0,937	0,946	696	all
Attach Box Label	600	0,910	0,900	0,905	291	all
Scan Label	700	0,966	0,943	0,954	788	all
Attach Shipping Label	800	0,939	0,924	0,931	447	all
Put on Back Table	900	0,911	0,935	0,923	262	all
Fill out Order	1000	0,935	0,920	0,927	499	all

Tabla 5.4: Resultados sobre giroscopio del sensor IMU de muñeca derecha

name	id	precision	recall	f1	support	key
avg/macro	-1	0,927	0,921	0,923		all
avg/weighted	-1	0,933	0,920	0,926		all
Picking	100	0,897	0,935	0,916	589	all
Relocate Item Label	200	0,965	0,854	0,906	881	all
Assemble Box	300	0,935	0,949	0,942	1023	all
Insert Items	400	0,930	0,892	0,910	489	all
Close Box	500	0,948	0,924	0,936	696	all
Attach Box Label	600	0,883	0,907	0,895	291	all
Scan Label	700	0,952	0,933	0,942	788	all
Attach Shipping Label	800	0,911	0,957	0,933	447	all
Put on Back Table	900	0,934	0,924	0,929	262	all
Fill out Order	1000	0,910	0,932	0,921	499	all

Tabla 5.5: Resultados sobre *keypoints* de huesos

name	id	precision	recall	f1	support	key
avg/macro	-1	0,934	0,928	0,931		all
avg/weighted	-1	0,938	0,933	0,935		all
Picking	100	0,935	0,959	0,947	589	all
Relocate Item Label	200	0,950	0,932	0,941	881	all
Assemble Box	300	0,923	0,947	0,935	1023	all
Insert Items	400	0,911	0,881	0,896	489	all
Close Box	500	0,952	0,947	0,950	696	all
Attach Box Label	600	0,897	0,893	0,895	291	all
Scan Label	700	0,969	0,937	0,952	788	all
Attach Shipping Label	800	0,937	0,935	0,936	447	all
Put on Back Table	900	0,937	0,912	0,925	262	all
Fill out Order	1000	0,930	0,932	0,931	499	all

Tabla 5.6: Resultados sobre *keypoints* de articulaciones

name	id	precision	recall	f1	support	key
avg/macro	-1	0,844	0,850	0,846		all
avg/weighted	-1	0,856	0,850	0,852		all
Picking	100	0,807	0,835	0,821	589	all
Relocate Item Label	200	0,887	0,807	0,845	881	all
Assemble Box	300	0,867	0,857	0,862	1023	all
Insert Items	400	0,764	0,869	0,813	489	all
Close Box	500	0,851	0,843	0,847	696	all
Attach Box Label	600	0,829	0,866	0,847	291	all
Scan Label	700	0,915	0,907	0,911	788	all
Attach Shipping Label	800	0,881	0,843	0,862	447	all
Put on Back Table	900	0,749	0,844	0,794	262	all
Fill out Order	1000	0,892	0,828	0,859	499	all

Tabla 5.7: Resultados sobre *keypoints* de huesos en movimiento

name	id	precision	recall	f1	support	key
avg/macro	-1	0,866	0,873	0,869		all
avg/weighted	-1	0,878	0,873	0,875		all
Picking	100	0,886	0,840	0,862	589	all
Relocate Item Label	200	0,914	0,832	0,871	881	all
Assemble Box	300	0,897	0,890	0,893	1023	all
Insert Items	400	0,840	0,877	0,858	489	all
Close Box	500	0,856	0,869	0,862	696	all
Attach Box Label	600	0,815	0,876	0,844	291	all
Scan Label	700	0,928	0,895	0,911	788	all
Attach Shipping Label	800	0,829	0,902	0,864	447	all
Put on Back Table	900	0,842	0,855	0,848	262	all
Fill out Order	1000	0,858	0,896	0,876	499	all

Tabla 5.8: Resultados sobre *keypoints* de articulaciones en movimiento

name	id	precision	recall	f1	support	key
avg/macro	-1	0,962	0,961	0,961		all
avg/weighted	-1	0,963	0,96	0,961		all
Picking	100	0,97	0,953	0,961	589	all
Relocate Item Label	200	0,965	0,954	0,959	881	all
Assemble Box	300	0,915	0,929	0,922	1023	all
Insert Items	400	0,99	0,921	0,954	489	all
Close Box	500	0,928	0,899	0,913	696	all
Attach Box Label	600	0,903	0,907	0,905	291	all
Scan Label	700	0,945	0,993	0,968	788	all
Attach Shipping Label	800	0,941	0,959	0,950	447	all
Put on Back Table	900	0,939	0,934	0,936	262	all
Fill out Order	1000	0,943	0,936	0,939	499	all

Tabla 5.9: Resultados sobre el modelo unificado

Los datos de movimiento son considerablemente peores, de todos modos, se mantienen por encima del 0.85 y analizaremos su aportación al modelo conjunto en el siguiente apartado. En los datos estáticos se obtienen resultados bastante similares a los calculados para DeepConvLSTM.

5.3. Resultados finales

En vista a los resultados anteriores se obtiene el resultado de la tabla 5.9 para el conjunto de evaluación en el modelo unificado:

Los modelos conjuntos han conseguido mejorías en todas las operaciones. Especialmente, hay que remarcar que gracias a la acción conjunta se consigue superar el 0.9 de valor-f para la actividad de colocar la etiqueta ya que la combinación de aciertos en distintas métricas ha conseguido compensar los errores de detección en una con los de la otra. En general este fenómeno se ha dado también para el resto de clases elevando precisión, exhaustividad y valor-f en 3 puntos destacando el éxito del uso de un modelo multi-modal.

Sobre el conjunto de entrega se actúa a ciegas, pues se desconocen las etiquetas reales y es el conjunto sobre el que se realiza la entrega del mejor modelo. El objetivo de este conjunto es evaluar sobre la entrega final, y al igual que los conjuntos de test en un proyecto de aprendizaje automático, no debe utilizarse para realizar el ajuste fino de hiperparámetros, pues para eso ya contamos con un conjunto de validación. No obstante, a modo de ir viendo el progreso a lo largo del trabajo se han realizado entregas realizadas con los modelos parciales de cara a ver las mejorías.

Estas entregas se hacen en la web de Codalab[1], donde existen dos formas de entrega, la *development* para comprobar los resultados durante el desarrollo y la final para la entrega que es válida para la participación en el concurso. Esta última está cerrada desde el fin del concurso, por lo que se

[1]https://codalab.lisn.upsaclay.fr/competitions/7830

RESULTS

#	User	Entries	Date of Last Entry	Team Name	f1 (macro avg.) ▲	f1 (weighted avg.) ▲	U0104 ▲	U0108 ▲	U0110 ▲	U0203 ▲	U0204 ▲	U0207 ▲
1	Malton	12	01/15/23	Malton	0.9846 (1)	0.9822 (1)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
2	tomoon	3	11/11/22		0.9578 (2)	0.9676 (2)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
3	yuto21	4	01/13/23	vbu211	0.9544 (3)	0.9617 (3)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
4	aalanderos95	8	01/07/23	Potros	0.9139 (4)	0.9293 (4)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
5	compound	5	12/25/22		0.9000 (5)	0.8990 (7)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
6	hmwri	29	12/12/22		0.8974 (6)	0.9113 (5)	0.9430 (1)	0.9220 (1)	0.8652 (2)	0.9101 (1)	0.8241 (1)	0.8156 (1)
7	aalanderos95	8	12/18/22		0.8951 (7)	0.9032 (6)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
8	XAVIER096	21	01/15/23	Dialga	0.8926 (8)	0.8887 (9)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
9	liuqijd	93	03/15/23	liuqijd	0.8892 (9)	0.8976 (8)	0.9128 (2)	0.9154 (2)	0.8853 (1)	0.9021 (2)	0.7886 (2)	0.8123 (2)
10	UD	5	01/14/23		0.8725 (10)	0.8809 (10)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
11	yuto21	4	10/31/22		0.8701 (11)	0.8753 (11)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
12	javierfuenca	7	07/06/23		0.8489 (12)	0.8642 (12)	0.8836 (3)	0.8925 (3)	0.8192 (3)	0.8663 (4)	0.7357 (3)	0.7764 (4)

Figura 5.1: Resultados de entrega del acelerómetro de la muñeca derecha sobre el conjunto de entrega

ha utilizado la entrega en modo *development*, que sigue abierta para poder seguir practicando aunque el concurso ya no esté activo. Hay que mencionar que la entrega tipo *development* no requiere realizar una inferencia sobre todo el conjunto, sino que con realizar una inferencia sobre la sesión U0102-S0500 es suficiente, sin embargo, también permite hacerlo evaluando sobre todo el conjunto simulando una participación real.

Para una primera entrega con únicamente datos del acelerómetro de la muñeca derecha se obtuvieron los resultados de la figura 5.1 que es una captura de la web donde se aprecia la posición 12 en valor-f pero hay que tener en cuenta que la mayoría de participantes ha subido únicamente los resultados de la inferencia de la sesión U0102-S0500. Puede verse que los resultados difieren de lo obtenido en los conjuntos de prueba. Esto es debido a que este conjunto de evaluación incluye una mayor cantidad de sesiones y que además estas contienen las variaciones en procedimientos (saltándose el orden entre operaciones, realizándolas de manera distinta, error que se discutirá más adelante, etc.) y no se había realizado aún una búsqueda óptima de hiper-parámetros.

Una vez probado el modelo para el acelerómetro de una muñeca se entrenaron ambas muñecas en IMU y se realizó una predicción conjunta aumentando considerablemente el valor-f tanto general como ponderada y subiendo hasta la posición 10 como puede verse en la figura 5.2. Al igual que el caso mencionado anteriormente, esta diferencia es debida a que no se había realizado el ajuste fino de hiper parámetros junto a las variabilidades mencionadas anteriormente.

RESULTS

#	User	Entries	Date of Last Entry	Team Name	f1 (macro avg.) ▲	f1 (weighted avg.) ▲	U0104 ▲	U0108 ▲	U0110 ▲	U0203 ▲	U0204 ▲	U0207 ▲
1	Malton	12	01/15/23	Malton	0.9846 (1)	0.9822 (1)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
2	tomoon	3	11/11/22		0.9578 (2)	0.9676 (2)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
3	yuto21	4	01/13/23	vbu211	0.9544 (3)	0.9617 (3)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
4	aalanderos95	8	01/07/23	Potros	0.9139 (4)	0.9293 (4)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
5	compound	5	12/25/22		0.9000 (5)	0.8990 (7)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
6	hmwri	29	12/12/22		0.8974 (6)	0.9113 (5)	0.9430 (1)	0.9220 (1)	0.8652 (3)	0.9101 (1)	0.8241 (2)	0.8156 (1)
7	aalanderos95	8	12/18/22		0.8951 (7)	0.9032 (6)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
8	XAVIER096	21	01/15/23	Dialga	0.8926 (8)	0.8887 (10)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
9	liuqijd	93	03/15/23	liuqijd	0.8892 (9)	0.8976 (9)	0.9128 (3)	0.9154 (2)	0.8853 (2)	0.9021 (2)	0.7886 (3)	0.8123 (2)
10	javierfuenca	9	07/11/23		0.8860 (10)	0.8980 (8)	0.9130 (2)	0.8986 (3)	0.8995 (1)	0.8856 (3)	0.8246 (1)	0.8021 (3)

Figura 5.2: Resultados de entrega del acelerómetro y giroscopio de ambas muñecas sobre el conjunto de entrega

Una vez entrenados los datos de *keypoints* estáticos y elaborando solución conjunta con los datos de IMUs se consiguió una mejora notable superando 0.9 de valor-f y llegando hasta una quinta posición como se observa en la figura 5.3. Aquí ya se contaba con una solución más acertada, pese a todo, siguen presentes las diferencias comentadas.

Finalmente, una vez añadidos los datos de los *keypoints* en movimiento se ve cómo mejoró ligeramente el resultado final afianzando un 5º puesto a fecha 11 de septiembre de 2023 que se ve en la figura 5.4 y que sobre la clasificación hubiera conseguido una tercera posición basándonos en el valor-f *macro average* y un cuarto puesto si miráramos el valor-f ponderado, como puede verse en el top de clasificaciones en entrega de competición en la figura 5.5.

En las figuras 5.6 y 5.7 se muestran ejemplos de dos sesiones de distintos usuarios donde se expresado la probabilidad de clase a lo largo del tiempo con las predicciones generadas por el modelo con la etiqueta o *groundtruth* superpuesta[2]. Se puede distinguir con claridad que el ajuste se corresponde con los resultados obtenidos siendo bastante preciso salvo en circunstancias en las que el operario ha realizado alguna variación de la actividad haciendo algo fuera delo recogido en el catálogo de clases (clase nula 10). También se puede ver que no todas las tareas tienen la misma duración y en alguna ocasión ciertas operaciones se extienden más en el tiempo o son más difíciles de detectar para el modelo como entre los segundos 970 y 1000 de la figura 5.6 y entre 220 y 240, entre 340 y 360 o 760 y 790 de la figura 5.7.

[2]Inspirado en el notebook de las guías oficiales de la web https://colab.research.google.com/github/open-pack/openpack-torch/blob/main/examples/unet/notebooks/U-Net_Change-Input-Data.ipynb

RESULTS

#	User	Entries	Date of Last Entry	Team Name	f1 (macro avg.) ▲	f1 (weighted avg.) ▲	U0104 ▲	U0108 ▲	U0110 ▲	U0203 ▲	U0204 ▲	U0207 ▲
1	Malton	12	01/15/23	Malton	0.9846 (1)	0.9822 (1)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
2	tomoon	3	11/11/22		0.9578 (2)	0.9676 (2)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
3	yuto21	4	01/13/23	vbu211	0.9544 (3)	0.9617 (3)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
4	liuqijd	93	04/06/23	liuqijd	0.9496 (4)	0.9508 (4)	0.9656 (1)	0.9602 (1)	0.9385 (1)	0.9635 (1)	0.8831 (1)	0.9265 (1)
5	javierfuenca	15	07/27/23		0.9189 (5)	0.9278 (6)	0.9268 (3)	0.9264 (2)	0.9249 (2)	0.9285 (2)	0.8614 (2)	0.8704 (2)

Figura 5.3: Resultados de entrega de datos IMU y *keypoints* sin movimiento sobre el conjunto de entrega

RESULTS

#	User	Entries	Date of Last Entry	Team Name	f1 (macro avg.) ▲	f1 (weighted avg.) ▲	U0104 ▲	U0108 ▲	U0110 ▲	U0203 ▲	U0204 ▲	U0207 ▲
1	Malton	13	01/15/23	Malton	0.9846 (1)	0.9822 (1)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
2	tomoon	3	11/11/22		0.9578 (2)	0.9676 (2)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
3	yuto21	4	01/13/23	vbu211	0.9544 (3)	0.9617 (3)	- (6)	- (6)	- (6)	- (6)	- (6)	- (6)
4	liuqijd	93	04/06/23	liuqijd	0.9496 (4)	0.9508 (4)	0.9656 (1)	0.9602 (1)	0.9385 (2)	0.9635 (1)	0.8831 (1)	0.9265 (1)
5	javierfuenca	26	08/15/23		0.9254 (5)	0.9352 (5)	0.9358 (3)	0.9304 (2)	0.9395 (1)	0.9287 (2)	0.8744 (2)	0.8740 (2)

Figura 5.4: Resultados de entrega de datos IMU y *keypoints* completos sobre el conjunto de entrega

RESULTS

#	User	Entries	Date of Last Entry	Team Name	f1 (macro avg.) ▲	f1 (weighted avg.) ▲	U0104 ▲	U0108 ▲	U0110 ▲	U0203 ▲	U0204 ▲	U0207 ▲
1	tomoon	61	01/16/23	tomoon	0.9633 (1)	0.9653 (1)	0.9757 (1)	0.9711 (1)	0.9660 (2)	0.9721 (1)	0.9138 (2)	0.9245 (1)
2	yuto21	68	01/14/23	vbu211	0.9592 (2)	0.9620 (2)	0.9671 (2)	0.9640 (2)	0.9705 (1)	0.9669 (2)	0.9155 (1)	0.9102 (2)
3	crazy112	65	01/06/23		0.9241 (3)	0.9361 (3)	0.9309 (6)	0.9297 (5)	0.9317 (3)	0.9317 (4)	0.8846 (3)	0.8747 (4)
4	crazy112	65	01/15/23	Ritsumei	0.9232 (4)	0.9350 (4)	0.9338 (5)	0.9256 (6)	0.9311 (4)	0.9306 (5)	0.8805 (4)	0.8763 (3)

Figura 5.5: Resultados de clasificación de la competición, donde basándonos en las métricas se obtendría 3º o 4º puesto contando valor-f macro o ponderado respectivamente

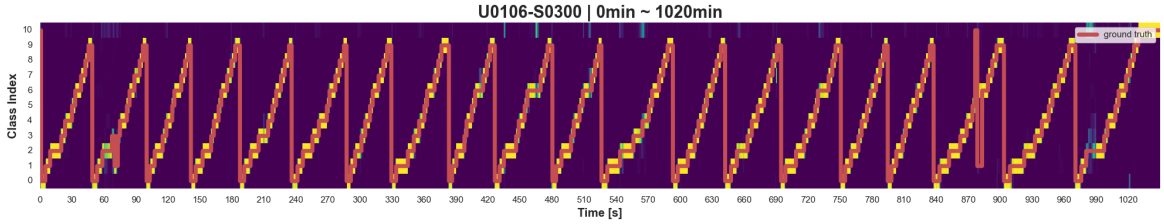


Figura 5.6: Predicción de la sesión 300 del usuario 106

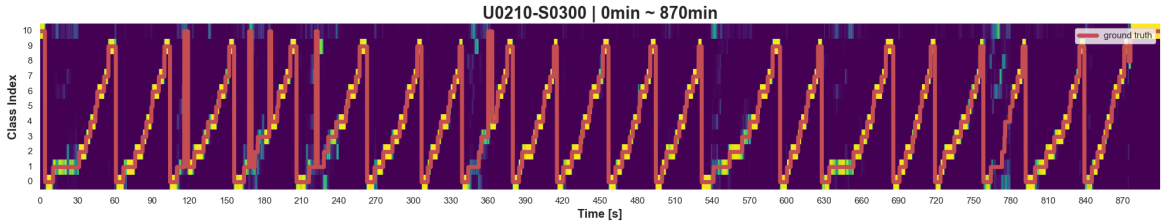


Figura 5.7: Predicción de la sesión 300 del usuario 210

5.4. Discusión y limitaciones

Se ha obtenido un resultado considerablemente preciso: alcanzar un valor-f superior a 0.9 ha es un hito bastante destacable, pero no solo eso sino que se ha entrado dentro de los 3 primeros modelos que originalmente se presentaron y entre los cinco presentados hasta la fecha (aunque 4 de ellos no cuentan con inferencia sobre todo el conjunto de entrega, solo sobre una sesión de un operario), por lo que de haber participado a tiempo se hubiera podido acceder al premio que era para los 5 primeros modelos. La notable mejora en los resultados de combinar las redes neuronales de sensores IMU con las de *keypoints* demuestra el éxito de la aproximación multi-modal. Además, añadir datos de movimiento como recomendaban (Liu et al., 2023) ha supuesto también una mejoría algo menos notoria.

Es esencial mencionar que, como se ha visto a lo largo de la exploración de los datos y la interpretación de los resultados, las operaciones tienen una fuerte correlación temporal y siguen un orden muy concreto salvo ciertos casos puntuales. Esto ha provocado un ligero sobre-entrenamiento que causa esas discrepancias entre resultados de validación/evaluación y entrega. Asimismo, algunas de las imprecisiones en los cálculos pueden solventarse mediante una capa de lógica que se aplique por encima para descartar falsos positivos como pueda ser el caso de que se detecte que un operario realiza alguna acción sobre la caja antes de haber detectado siquiera que la haya recogido previamente. El sistema presenta resultados peores para los usuarios U204 y U207, donde se indica en la

documentación [3] el problema de que el sensor inercial correspondiente a la muñeca izquierda estaba colocado en sentido contrario y precisa un tratamiento para reconstruir los datos. Estos datos están siendo interpretados erróneamente y esa variabilidad induce al error en la detección que hemos ido comentando a lo largo de todos los resultados.

En el caso de que se quisiera implementar esta solución en tiempo real, en cuanto a tiempos de inferencia no se han encontrado problemas, pues el tiempo de procesamiento es suficientemente rápido como para ofrecer una respuesta cercana al tiempo real. Sin embargo, habría que añadir la sobrecarga que supone el realizar una extracción de la pose, aunque esta es posible obtenerla en tiempo real actualmente, pero conviene estimar este *overhead* que puede traer.

A pesar de todo, se ha contado con una serie de limitaciones que serán analizadas a continuación:

- El tiempo de realización de este trabajo es limitado y debe ceñirse a las 300 horas aproximadas estipuladas en la guía de la asignatura donde de acuerdo a la estimación realizada en la propuesta de este trabajo no todas pueden invertirse en la investigación y desarrollo de mejores modelos. En las líneas futuras se comentarán otras aproximaciones y adiciones que serían posibles si se dispusiera de más tiempo.

- El conjunto de datos está en constante evolución y cada cierto tiempo se presenta una nueva versión corrigiendo errores pasados y mejorando la calidad de las etiquetas. En un futuro no muy lejano se puede aprovechar este enriquecimiento progresivo para entrenar sobre datos de mayor calidad e incluso utilizando sensores no usados en esta práctica debido a los problemas de recolección y etiquetado expuestos en la sección 2.3.

- El hardware con el que se cuenta era un ordenador portátil personal con altas prestaciones, pero que podrían ser incluso mayores si se hubiera tenido acceso a un proveedor *cloud* con máquinas dedicadas y con hardware más específico (en especial GPUs especializadas en *deep learning*) para poder realizar más entrenamientos en paralelo sin depender de dejar horas o incluso días enteros el ordenador inutilizado durante entrenamientos.

[3] https://github.com/open-pack/openpack-toolkit/blob/main/docs/USER.md

Capítulo 6

Conclusiones y trabajos futuros

6.1. Conclusiones

En este trabajo se ha desarrollado un modelo multi-modal de clasificación de acciones a partir de un ambicioso *dataset* que abarca múltiples disciplinas del tratamiento de datos debido a su amplia gama de sensores. A pesar de las limitaciones en tiempo, se ha conseguido entrenar un sistema que ha permitido entrar entre los 5 primeros participantes del concurso. Esto indica que si se hubiera participado se habría entrado entre los ganadores, como indican en su web los 5 primeros modelos recibieron un premio. Si bien es cierto, el modelo de keypoints utilizado ha sido publicado este mismo año y no ha habido tiempo suficiente para entrenarlo y participar, aunque esto último escapa del foco principal del trabajo

Poniendo el foco en el objetivo, contar con los resultados de otros participantes ha permitido realizar una comparativa en la que se ha demostrado el éxito del modelo desarrollado. Se ha conseguido a su vez profundizar en técnicas de aprendizaje automático, y más en concreto, de aprendizaje profundo correspondientes con asignaturas de este máster. Se han implementado modelos conjuntos, vistos en aprendizaje automático II. Las estructuras utilizadas para la red neuronal de los sensores IMU eran conocidas en estas asignaturas, pero se ha aprovechado las ventajas de ambas para usar su combinación. Por el lado de los datos de cámaras y visión, la extracción de la pose ha sido uno de los principales componentes y, a pesar de venir dado, se ha estudiado su funcionamiento que cuenta con cierta madurez. Además, se ha explotado una técnica muy reciente de reconocimiento de actividades con las GCN con las estructuras de los grafos, cuya estructura subyacente, el grafo, ha sido tratado en múltiples asignaturas (ICPDM, GAINE, MDMS) ajustándolo al contexto del aprendizaje profundo.

La curva de aprendizaje se ha visto suavizada gracias a las herramientas con las que se ha contado proporcionadas por los autores del dataset como el *toolkit* y la documentación, pues han sido completos e indispensables para la correcta realización de la tarea. El propio marco de trabajo se ha desarrollado con PyTorch, una alternativa a Tensorflow y que está en auge en los trabajos de investigación.

6.2. Trabajos futuros

Este trabajo puede llegar a requerir un gran número de horas que se han dosificado lo más compactamente posible ajustándolo al tiempo disponible y permitiendo acabar con una entrega satisfactoria. Aún así, este conjunto de datos abre las puertas a un gran catálogo de mejoras y pruebas. Dentro todas ellas se van a destacar las principales y que podrían ser objeto de interés en futuras investigaciones:

- Explotar los datos provenientes de otros sensores: A pesar de que algunos pueden presentar problemas para su tratamiento (las nubes de puntos de LiDAR son muy complejas), sería de gran interés explorar datos como los de las cámaras de profundidad o de los otros sensores. Como se explicó en el apartado 2.3, se optó por realizar el trabajo con determinados sensores pero eso no impide ver qué puede llegar a conseguirse con el resto. Según anuncian en su web[1], desde el 8 de julio están preparando la versión 1.0.0 del conjunto de datos corrigiendo bastantes de los errores que nos hemos encontrado previamente.

- Este conjunto de datos cuenta con unos metadatos enriquecidos que aportan gran información y pueden explotarse en mayor medida especialmente para resolver los sobre-entrenos mencionados debido a la correlación temporal a partir de muestreos aleatorios en determinadas etapas si abandonar la inherente relacción de consecución temporal. Introducir de alguna manera a los modelos datos de factores como la mano dominante, edad, experiencia o escenario de los descritos en la sección 2.1 en cada caso, podría acarrear mejoras a la hora de realizar la clasificación.

- Ajuste de hiperparámetros con mayor barrido: Dada la cantidad de parámetros configurables en este modelo, por motivos de tiempo se han realizados barridos con *GridSeach* de ajuste de hiperparámetros limitados, pues cada entrenamiento conllevaba procesos en ocasiones superiores a las 14 horas. Contando con un hardware más potente y con más tiempo sería de interés descubrir hasta dónde se puede llegar con la solución propuesta, especialmente al haber logrado situarse entre el puesto 3 y 4 de la clasificación competitiva.

- Otro punto que podría realizarse sería plantear un modelo de mayor precisión para el sensor IMU, pues DeepConvLSTM es muy eficaz, aunque data de 2016 y existen soluciones que, debido a ajustes de tiempos no se han contemplado, con alta probabilidad puedan ofrecer mejor rendimiento para la problemática de este trabajo.

[1] https://open-pack.github.io/release/v1-0-0

Bibliografía

Agarwal, P. and Alam, M. (2020). A lightweight deep learning model for human activity recognition on edge devices. *Procedia Computer Science*, 167:2364–2373.

Aggarwal, J. K. and Ryoo, M. S. (2011). Human activity analysis: A review. *Acm Computing Surveys (Csur)*, 43(3):1–43.

Ahmad, T., Jin, L., Zhang, X., Lai, S., Tang, G., and Lin, L. (2021). Graph convolutional neural network for human action recognition: A comprehensive survey. *IEEE Transactions on Artificial Intelligence*, 2(2):128–145.

Ascioglu, G. and Senol, Y. (2020). Design of a wearable wireless multi-sensor monitoring system and application for activity recognition using deep learning. *IEEE Access*, 8:169183–169195.

Ashry, S., Elbasiony, R., and Gomaa, W. (2018). An lstm-based descriptor for human activities recognition using imu sensors. In *Proceedings of the 15th International Conference on Informatics in Control, Automation and Robotics, ICINCO*, volume 1, pages 494–501.

Ashry, S., Ogawa, T., and Gomaa, W. (2020). Charm-deep: continuous human activity recognition model based on deep neural network using imu sensors of smartwatch. *IEEE Sensors Journal*, 20(15):8757–8770.

Atienza, R. O. (2019). Deep learning for smartphone-based human activity recognition using multi-sensor fusion. In *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Proceedings of the Wireless Internet: 11th EAI International Conference, WiCON 2018, Taipei, Taiwan, 15–16 October 2018*, page 65. Springer.

Avilés-Cruz, C., Ferreyra-Ramírez, A., Zúñiga-López, A., and Villegas-Cortéz, J. (2019). Coarse-fine convolutional deep-learning strategy for human activity recognition. *Sensors*, 19(7):1556.

Bachmann, E. R., Duman, I., Usta, U. Y., McGhee, R. B., Yun, X., and Zyda, M. (1999). Orientation tracking for humans and robots using inertial sensors. In *Proceedings 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation. CIRA'99 (Cat. No. 99EX375)*, pages 187–194. IEEE.

Badawi, A. A., Al-Kabbany, A., and Shaban, H. (2018). Multimodal human activity recognition from wearable inertial sensors using machine learning. In *2018 IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES)*, pages 402–407. IEEE.

Bernardin, K. and Stiefelhagen, R. (2008). Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10.

Bhat, G., Tran, N., Shill, H., and Ogras, U. Y. (2020). w-har: An activity recognition dataset and framework using low-power wearable devices. *Sensors*, 20(18):5356.

Blunsom, P. (2004). Maximum entropy markov models for semantic role labelling. In *Proceedings of the Australasian Language Technology Workshop 2004*, pages 109–116.

Bolme, D. S., Beveridge, J. R., Draper, B. A., and Lui, Y. M. (2010). Visual object tracking using adaptive correlation filters. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2544–2550. IEEE.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42.

Cardoso, A., Leitão, J., and Teixeira, C. (2019). Using the jupyter notebook as a tool to support the teaching and learning processes in engineering courses. In *The Challenges of the Digital Transformation in Education: Proceedings of the 21st International Conference on Interactive Collaborative Learning (ICL2018)-Volume 2*, pages 227–236. Springer.

Carreira, J. and Zisserman, A. (2017). Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308.

Chang, J.-W. and Liu, H.-R. (2023). Applying 5pkc-based skeleton partition strategy into spatio-temporal graph convolution networks for fitness action recognition. In *International Conference on Innovative Computing*, pages 728–737. Springer.

Chen, X., Guo, H., Wang, G., and Zhang, L. (2017). Motion feature augmented recurrent neural network for skeleton-based dynamic hand gesture recognition. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 2881–2885. IEEE.

Chen, Y. and Xue, Y. (2015). A deep learning approach to human activity recognition based on single accelerometer. In *2015 IEEE international conference on systems, man, and cybernetics*, pages 1488–1492. IEEE.

Chen, Y., Zhang, Z., Yuan, C., Li, B., Deng, Y., and Hu, W. (2021). Channel-wise topology refinement graph convolution for skeleton-based action recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 13359–13368.

Cheng, F.-C., Huang, S.-C., and Ruan, S.-J. (2010). Scene analysis for object detection in advanced surveillance systems using laplacian distribution model. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 41(5):589–598.

Cheng, X., Zhang, L., Tang, Y., Liu, Y., Wu, H., and He, J. (2022). Real-time human activity recognition using conditionally parametrized convolutions on mobile and wearable devices. *IEEE Sensors Journal*, 22(6):5889–5901.

Chung, S., Lim, J., Noh, K. J., Kim, G., and Jeong, H. (2019). Sensor data acquisition and multimodal sensor fusion for human activity recognition using deep learning. *Sensors*, 19(7):1716.

Cruciani, F., Vafeiadis, A., Nugent, C., Cleland, I., McCullagh, P., Votis, K., Giakoumis, D., Tzovaras, D., Chen, L., and Hamzaoui, R. (2020). Feature learning for human activity recognition using convolutional neural networks: A case study for inertial measurement unit and audio data. *CCF Transactions on Pervasive Computing and Interaction*, 2(1):18–32.

Cucchiara, R., Grana, C., Piccardi, M., and Prati, A. (2003). Detecting moving objects, ghosts, and shadows in video streams. *IEEE transactions on pattern analysis and machine intelligence*, 25(10):1337–1342.

Denman, S., Fookes, C., and Sridharan, S. (2009). Improved simultaneous computation of motion detection and optical flow for object tracking. In *2009 Digital Image Computing: Techniques and Applications*, pages 175–182. IEEE.

Diete, A. and Stuckenschmidt, H. (2019). Fusing object information and inertial data for activity recognition. *Sensors*, 19(19):4119.

Diete, A., Sztyler, T., and Stuckenschmidt, H. (2019). Vision and acceleration modalities: Partners for recognizing complex activities. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 101–106. IEEE.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634.

Dong, J., Gao, Y., Lee, H. J., Zhou, H., Yao, Y., Fang, Z., and Huang, B. (2020). Action recognition based on the fusion of graph convolutional networks with high order features. *Applied Sciences*, 10(4):1482.

Du, W., Wang, Y., and Qiao, Y. (2017). Rpan: An end-to-end recurrent pose-attention network for action recognition in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 3725–3734.

Farha, Y. A. and Gall, J. (2019). Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3575–3584.

Gao, J., Zhang, T., and Xu, C. (2019). I know the relationships: Zero-shot action recognition via two-stream graph convolutional networks and knowledge graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 8303–8311.

Gao, J., Zhang, T., and Xu, C. (2020). Learning to model relationships for zero-shot video classification. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3476–3491.

Gao, W., Zhang, L., Teng, Q., He, J., and Wu, H. (2021). Danhar: Dual attention network for multimodal human activity recognition using wearable sensors. *Applied Soft Computing*, 111:107728.

Gavrila, D. M. (1999). The visual analysis of human movement: A survey. *Computer vision and image understanding*, 73(1):82–98.

Gomaa, W., Elbasiony, R., and Ashry, S. (2017). Adl classification based on autocorrelation function of inertial signals. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 833–837. IEEE.

Gumaei, A., Hassan, M. M., Alelaiwi, A., and Alsalman, H. (2019). A hybrid deep learning model for human activity recognition using multimodal body sensing data. *IEEE Access*, 7:99152–99160.

Hamilton, W. L., Ying, R., and Leskovec, J. (2017). Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*.

He, J., Zhang, Q., Wang, L., and Pei, L. (2018). Weakly supervised human activity recognition from wearable sensors by recurrent attention learning. *IEEE Sensors Journal*, 19(6):2287–2297.

Hu, J.-F., Zheng, W.-S., Lai, J., and Zhang, J. (2015). Jointly learning heterogeneous features for rgb-d activity recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5344–5352.

Huang, S.-C. (2010). An advanced motion detection algorithm with video quality analysis for video surveillance systems. *IEEE transactions on circuits and systems for video technology*, 21(1):1–14.

Hussein, M. E., Torki, M., Gowayyed, M. A., and El-Saban, M. (2013). Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. In *Twenty-third international joint conference on artificial intelligence*.

Ignatov, A. (2018). Real-time human activity recognition from accelerometer data using convolutional neural networks. *Applied Soft Computing*, 62:915–922.

Ince, S. and Konrad, J. (2008). Occlusion-aware optical flow estimation. *IEEE Transactions on Image Processing*, 17(8):1443–1451.

Jeong, C. Y. and Kim, M. (2019). An energy-efficient method for human activity recognition with segment-level change detection and deep learning. *Sensors*, 19(17):3688.

Kalabakov, S., Gjoreski, M., Gjoreski, H., and Gams, M. (2021). Analysis of deep transfer learning using deepconvlstm for human activity recognition from wearable sensors. *Informatica*, 45(2).

Karpathy, A., Johnson, J., and Fei-Fei, L. (2015). Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*.

Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al. (2017). The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*.

Kerber, F., Puhl, M., and Krüger, A. (2017). User-independent real-time hand gesture recognition based on surface electromyography. In *Proceedings of the 19th international conference on human-computer interaction with mobile devices and services*, pages 1–7.

Khan, M. A. A. H., Roy, N., and Misra, A. (2018). Scaling human activity recognition via deep learning-based domain adaptation. In *2018 IEEE international conference on pervasive computing and communications (PerCom)*, pages 1–9. IEEE.

Kipf, T. N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Laptev, I. (2005). On space-time interest points. *International journal of computer vision*, 64:107–123.

Lee, J. B., Rossi, R. A., Kim, S., Ahmed, N. K., and Koh, E. (2019). Attention models in graphs: A survey. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(6):1–25.

Lee, S. (2012). Depth camera image processing and applications. In *2012 19th IEEE International Conference on Image Processing*, pages 545–548. IEEE.

Li, B., Li, X., Zhang, Z., and Wu, F. (2019). Spatio-temporal graph routing for skeleton-based action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8561–8568.

Li, C., Hou, Y., Wang, P., and Li, W. (2017). Joint distance maps based action recognition with convolutional neural networks. *IEEE Signal Processing Letters*, 24(5):624–628.

Li, G., Yang, S., and Li, J. (2020a). Edge and node graph convolutional neural network for human action recognition. In *2020 Chinese Control And Decision Conference (CCDC)*, pages 4630–4635. IEEE.

Li, S., Jiang, T., Huang, T., and Tian, Y. (2020b). Global co-occurrence feature learning and active coordinate system conversion for skeleton-based action recognition. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 586–594.

Li, W., Zhang, Z., and Liu, Z. (2010). Action recognition based on a bag of 3d points. In *2010 IEEE computer society conference on computer vision and pattern recognition-workshops*, pages 9–14. IEEE.

Lillo, I., Soto, A., and Carlos Niebles, J. (2014). Discriminative hierarchical modeling of spatio-temporally composable human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 812–819.

Liu, D., Chen, P., Yao, M., Lu, Y., Cai, Z., and Tian, Y. (2023). Tsgcnext: Dynamic-static multi-graph convolution for efficient skeleton-based action recognition with long-term learning potential. *arXiv preprint arXiv:2304.11631*.

Liu, J., Shahroudy, A., Perez, M., Wang, G., Duan, L.-Y., and Kot, A. C. (2019). Ntu rgb+ d 120: A large-scale benchmark for 3d human activity understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2684–2701.

Liu, M. and Yuan, J. (2018). Recognizing human actions as the evolution of pose estimation maps. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1159–1168.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. (2022). A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986.

Luo, W., Zhang, C., Zhang, X., and Wu, H. (2019). Improving action recognition with the graph-neural-network-based interaction reasoning. In *2019 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4. IEEE.

Lv, T., Wang, X., Jin, L., Xiao, Y., and Song, M. (2020). Margin-based deep learning networks for human activity recognition. *Sensors*, 20(7):1871.

Moeslund, T. B., Hilton, A., and Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding*, 104(2-3):90–126.

Morales, J., Yoshimura, N., Xia, Q., Wada, A., Namioka, Y., and Maekawa, T. (2022). Acceleration-based human activity recognition of packaging tasks using motif-guided attention networks. In

2022 IEEE international conference on pervasive computing and communications (PerCom), pages 1–12. IEEE.

Morris, B. T. and Trivedi, M. M. (2008). A survey of vision-based trajectory learning and analysis for surveillance. *IEEE transactions on circuits and systems for video technology*, 18(8):1114–1127.

Mukherjee, D., Mondal, R., Singh, P. K., Sarkar, R., and Bhattacharjee, D. (2020). Ensemconvnet: a deep learning approach for human activity recognition using smartphone sensors for healthcare applications. *Multimedia Tools and Applications*, 79:31663–31690.

Niemann, F., Reining, C., Moya Rueda, F., Nair, N. R., Steffens, J. A., Fink, G. A., and Ten Hompel, M. (2020). Lara: Creating a dataset for human activity recognition in logistics using semantic attributes. *Sensors*, 20(15):4083.

Nunez, J. C., Cabido, R., Pantrigo, J. J., Montemayor, A. S., and Velez, J. F. (2018). Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition. *Pattern Recognition*, 76:80–94.

Nutter, M., Crawford, C. H., and Ortiz, J. (2018). Design of novel deep learning models for real-time human activity recognition with mobile phones. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Oral, M. and Deniz, U. (2007). Centre of mass model–a novel approach to background modelling for segmentation of moving objects. *Image and vision computing*, 25(8):1365–1376.

Ordóñez, F. J. and Roggen, D. (2016). Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115.

Pham, C., Nguyen-Thai, S., Tran-Quang, H., Tran, S., Vu, H., Tran, T.-H., and Le, T.-L. (2020). Senscapsnet: deep neural network for non-obtrusive sensing based human activity recognition. *IEEE Access*, 8:86934–86946.

Poppe, R. (2010). A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990.

Qi, W., Su, H., and Aliverti, A. (2020). A smartphone-based adaptive recognition and real-time monitoring system for human activities. *IEEE Transactions on Human-Machine Systems*, 50(5):414–423.

Qingxin, X., Wada, A., Korpela, J., Maekawa, T., and Namioka, Y. (2019). Unsupervised factory activity recognition with wearable sensors using process instruction information. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(2):1–23.

Rashid, N., Demirel, B. U., and Al Faruque, M. A. (2022). Ahar: Adaptive cnn for energy-efficient human activity recognition in low-power edge devices. *IEEE Internet of Things Journal*, 9(15):13041–13051.

Ronao, C. A. and Cho, S.-B. (2016). Human activity recognition with smartphone sensors using deep learning neural networks. *Expert systems with applications*, 59:235–244.

Rueda, F. M., Lüdtke, S., Schröder, M., Yordanova, K., Kirste, T., and Fink, G. A. (2019). Combining symbolic reasoning and deep learning for human activity recognition. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 22–27. IEEE.

Sargano, A. B., Angelov, P., and Habib, Z. (2017). A comprehensive review on handcrafted and learning-based action representation approaches for human activity recognition. *applied sciences*, 7(1):110.

Shahroudy, A., Liu, J., Ng, T.-T., and Wang, G. (2016). Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1010–1019.

Shi, L., Zhang, Y., Cheng, J., and Lu, H. (2019a). Skeleton-based action recognition with directed graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7912–7921.

Shi, L., Zhang, Y., Cheng, J., and Lu, H. (2019b). Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12026–12035.

Simonyan, K. and Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. *Advances in neural information processing systems*, 27.

Singh, S. P., Sharma, M. K., Lay-Ekuakille, A., Gangwar, D., and Gupta, S. (2020). Deep convlstm with self-attention for human activity decoding using wearable sensors. *IEEE Sensors Journal*, 21(6):8575–8582.

Su, T., Sun, H., Ma, C., Jiang, L., and Xu, T. (2019). Hdl: Hierarchical deep learning model based human activity recognition using smartphone sensors. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Sun, J., Fu, Y., Li, S., He, J., Xu, C., and Tan, L. (2018a). Sequential human activity recognition based on deep convolutional network and extreme learning machine using wearable sensors. *Journal of Sensors*, 2018.

Sun, K., Xiao, B., Liu, D., and Wang, J. (2019). Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5693–5703.

Sun, S., Kuang, Z., Sheng, L., Ouyang, W., and Zhang, W. (2018b). Optical flow guided feature: A fast and robust motion representation for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1390–1399.

Tang, C. I., Perez-Pozuelo, I., Spathis, D., Brage, S., Wareham, N., and Mascolo, C. (2021). Selfhar: Improving human activity recognition through self-training with unlabeled data. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies*, 5(1):1–30.

Tang, Y., Teng, Q., Zhang, L., Min, F., and He, J. (2020). Layer-wise training convolutional neural networks with smaller filters for human activity recognition using wearable sensors. *IEEE Sensors Journal*, 21(1):581–592.

Toshev, A. and Szegedy, C. (2014). Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660.

Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. (2015). Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497.

Turaga, P., Chellappa, R., Subrahmanian, V. S., and Udrea, O. (2008). Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video technology*, 18(11):1473–1488.

Vishwakarma, S. and Agrawal, A. (2013). A survey on activity recognition and behavior understanding in video surveillance. *The Visual Computer*, 29:983–1009.

Vrigkas, M., Nikou, C., and Kakadiaris, I. A. (2015). A review of human activity recognition methods. *Frontiers in Robotics and AI*, 2:28.

Wan, S., Qi, L., Xu, X., Tong, C., and Gu, Z. (2020). Deep learning models for real-time human activity recognition with smartphones. *Mobile Networks and Applications*, 25:743–755.

Wang, H., Zhao, J., Li, J., Tian, L., Tu, P., Cao, T., An, Y., Wang, K., and Li, S. (2020). Wearable sensor-based human activity recognition using hybrid deep learning techniques. *Security and communication Networks*, 2020:1–12.

Wang, J., Chen, Y., Hao, S., Peng, X., and Hu, L. (2019a). Deep learning for sensor-based activity recognition: A survey. *Pattern recognition letters*, 119:3–11.

Wang, J., Nie, X., Xia, Y., Wu, Y., and Zhu, S.-C. (2014). Cross-view action modeling, learning and recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2649–2656.

Wang, K., He, J., and Zhang, L. (2019b). Attention-based convolutional neural network for weakly labeled human activities' recognition with wearable sensors. *IEEE Sensors Journal*, 19(17):7598–7604.

Wang, K., He, J., and Zhang, L. (2021). Sequential weakly labeled multiactivity localization and recognition on wearable sensors using recurrent attention networks. *IEEE Transactions on Human-Machine Systems*, 51(4):355–364.

Wang, L. and Liu, R. (2020). Human activity recognition based on wearable sensor using hierarchical deep lstm networks. *Circuits, Systems, and Signal Processing*, 39:837–856.

Wang, L., Xiong, Y., Wang, Z., and Qiao, Y. (2015). Towards good practices for very deep two-stream convnets. *arXiv preprint arXiv:1507.02159*.

Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., and Van Gool, L. (2016). Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer.

Wang, S., Ma, Z., Yang, Y., Li, X., Pang, C., and Hauptmann, A. G. (2013). Semi-supervised multiple feature analysis for action recognition. *IEEE transactions on multimedia*, 16(2):289–298.

Wei, H., Chopada, P., and Kehtarnavaz, N. (2020). C-mhad: Continuous multimodal human action dataset of simultaneous video and inertial sensing. *Sensors*, 20(10):2905.

Wei, H., Jafari, R., and Kehtarnavaz, N. (2019). Fusion of video and inertial sensing for deep learning–based human action recognition. *Sensors*, 19(17):3680.

Wei, H. and Kehtarnavaz, N. (2020). Simultaneous utilization of inertial and video sensing for action detection and recognition in continuous action streams. *IEEE Sensors Journal*, 20(11):6055–6063.

Wei, S.-E., Ramakrishna, V., Kanade, T., and Sheikh, Y. (2016). Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4724–4732.

Wen, L., Nie, M., Chen, P., Zhao, Y.-n., Shen, J., Wang, C., Xiong, Y., Yin, K., and Sun, L. (2022). Wearable multimode sensor with a seamless integrated structure for recognition of different joint motion states with the assistance of a deep learning algorithm. *Microsystems & nanoengineering*, 8(1):24.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y. (2020). A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24.

Xia, K., Huang, J., and Wang, H. (2020a). Lstm-cnn architecture for human activity recognition. *IEEE Access*, 8:56855–56866.

Xia, Q., Korpela, J., Namioka, Y., and Maekawa, T. (2020b). Robust unsupervised factory activity recognition with body-worn accelerometer using temporal structure of multiple sensor data motifs. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(3):1–30.

Xiao, F., Pei, L., Chu, L., Zou, D., Yu, W., Zhu, Y., and Li, T. (2021). A deep learning method for complex human activity recognition using virtual wearable sensors. In *Spatial Data and Intelligence: First International Conference, SpatialDI 2020, Virtual Event, May 8–9, 2020, Proceedings 1*, pages 261–270. Springer.

Xu, C., Chai, D., He, J., Zhang, X., and Duan, S. (2019). Innohar: A deep neural network for complex human activity recognition. *Ieee Access*, 7:9893–9902.

Xu, J., Tasaka, K., and Yanagihara, H. (2018). Beyond two-stream: Skeleton-based three-stream networks for action recognition in videos. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1567–1573. IEEE.

Yan, S., Xiong, Y., and Lin, D. (2018). Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Yilmaz, A., Li, X., and Shah, M. (2004). Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions on pattern analysis and machine intelligence*, 26(11):1531–1536.

Yoshimura, N., Maekawa, T., Hara, T., Wada, A., and Namioka, Y. (2022a). Acceleration-based activity recognition of repetitive works with lightweight ordered-work segmentation network. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(2):1–39.

Yoshimura, N., Morales, J., Maekawa, T., and Hara, T. (2022b). Openpack: A large-scale dataset for recognizing packaging works in iot-enabled logistic environments. *arXiv preprint arXiv:2212.11152*.

Zhang, P., Lan, C., Xing, J., Zeng, W., Xue, J., and Zheng, N. (2019a). View adaptive neural networks for high performance skeleton-based human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1963–1978.

Zhang, S., Tong, H., Xu, J., and Maciejewski, R. (2019b). Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23.

Zhang, X., Angeloudis, P., and Demiris, Y. (2022). St crossingpose: A spatial-temporal graph convolutional network for skeleton-based pedestrian crossing intention prediction. *IEEE Transactions on Intelligent Transportation Systems*, 23(11):20773–20782.

Zhang, Z., Cui, P., and Zhu, W. (2020). Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):249–270.

Zhao, Y., Yang, R., Chevalier, G., Xu, X., and Zhang, Z. (2018). Deep residual bidir-lstm for human activity recognition using wearable sensors. *Mathematical Problems in Engineering*, 2018:1–13.

Zhen, X., Shao, L., Maybank, S. J., and Chellappa, R. (2016). Handcrafted vs. learned representations for human action recognition. *Image and Vision Computing*, 55(2):39–41.

Zhou, X., Liang, W., Kevin, I., Wang, K., Wang, H., Yang, L. T., and Jin, Q. (2020). Deep-learning-enhanced human activity recognition for internet of healthcare things. *IEEE Internet of Things Journal*, 7(7):6429–6438.

Zhu, Q., Chen, Z., and Soh, Y. C. (2018). A novel semisupervised deep learning method for human activity recognition. *IEEE Transactions on Industrial Informatics*, 15(7):3821–3830.

Zhu, R., Xiao, Z., Li, Y., Yang, M., Tan, Y., Zhou, L., Lin, S., and Wen, H. (2019). Efficient human activity recognition solving the confusing activities via deep ensemble learning. *Ieee Access*, 7:75490–75499.

Anexo A

Análisis Exploratorio de Datos

En este apéndice se puede observar el notebook empleado para realizar el análisis exploratorio de datos.

EDA

July 25, 2023

1 Análisis exploratorio de datos

En este notebook vamos a realizar una exploración del dataset de openpack de cara a saber con qué información se cuenta. Esta información será analizada y completada en el documento del trabajo principal, sirviendo el presente cuaderno de trabajo como apoyo de código a este.

Las principales librerías que se usarán serán pandas para la lectura de los ficheros y matplotlib para reperesentaciones. También se utilizará OpenCV para la visualización del esqueleto y Open3D para las nubes de puntos del LiDAR. Para ello nos apoyaremos en la librería openpack-toolkit, que a partir de una configuración OmegaConf nos permitirá fácilmente movernos por el contenido del dataset.

En total se van a analizar los siguientes elementos: * Etiquetas * Datos de sensores IMU: * Acelerómetro * Giroscopio * Cuaternión * Datos de sensores E4: * Acelerómetro * Medidor de presión * Actividad electrodérmica (EDA) * Temperatura * Puntos corporales (*keypoints*) extraídos de sensor Kinect * Datos del sistema: * Lector de etiquetas * Registro de pedidos * Impresora * Nubes de puntos de sensor LiDAR

Finalmente se discutirá en el documento la viabilidad de usar cada uno

[1]:
```
from pathlib import Path

import numpy as np
import pandas as pd
import json
import openpack_toolkit as optk
from omegaconf import DictConfig, OmegaConf, open_dict
from openpack_toolkit.utils.notebook import noglobal

%matplotlib inline
import matplotlib.pyplot as plt
from matplotlib.gridspec import GridSpec, GridSpecFromSubplotSpec
import seaborn as sns
import open3d as o3d
import cv2
sns.set("notebook", "whitegrid", font_scale=1.5)
```

```
Jupyter environment detected. Enabling Open3D WebVisualizer.
[Open3D INFO] WebRTC GUI backend enabled.
[Open3D INFO] WebRTCWindowSystem: HTTP handshake server disabled.
```

Partimos de todo el dataset en su versión v0.3.1 descargada en la ruta local data/datasets. Creamos un diccionario de configuración OmegaConf con los datos de usuario a utilizar, sesión (al menos hay que fijarla en None para que la clave esté creada), la ruta del dataset y la información sobre ese mismo dataset que será completada más adelante.

[2]:
```
cfg = OmegaConf.create({
    "user": optk.configs.users.U0101,
    "session": None,
    "path": {
        "openpack": {
            "version": "v0.3.1",
            "rootdir":  "data/datasets/openpack/v0.3.1",
        },
    },
    "dataset": {
        "annotation": None,
        "stream": None,
    }
})
print(OmegaConf.to_yaml(cfg))
```

```
user:
  id: 101
  name: U0101
  sessions:
    S0100:
      duration: 35m36s
      end: '2021-10-14T12:01:10+09:00'
      start: '2021-10-14T11:25:34+09:00'
    S0200:
      duration: 28m54s
      end: '2021-10-14T13:46:36+09:00'
      start: '2021-10-14T13:17:42+09:00'
    S0300:
      duration: 28m58s
      end: '2021-10-14T14:38:08+09:00'
      start: '2021-10-14T14:09:10+09:00'
    S0400:
      duration: 30m55s
      end: '2021-10-14T15:30:43+09:00'
      start: '2021-10-14T14:59:48+09:00'
    S0500:
      duration: 27m35s
      end: '2021-10-14T16:21:40+09:00'
      start: '2021-10-14T15:54:05+09:00'
session: null
path:
  openpack:
```

```
    version: v0.3.1
    rootdir: data/datasets/openpack/v0.3.1
dataset:
  annotation: null
  stream: null
```

1.1 Etiquetas

El reto de Openpack consiste en clasificar un total de 10 operaciones que se subdividen en acciones existiendo una extra que será descartable para abarcar los casos de etiqueta nula. A continuación, se muestran todas ellas junto a sus códigos.

[3]:
```
print(OmegaConf.to_yaml(optk.OPENPACK_OPERATIONS))
```

```
classes:
- id: 100
  name: Picking
  is_ignore: false
- id: 200
  name: Relocate Item Label
  is_ignore: false
- id: 300
  name: Assemble Box
  is_ignore: false
- id: 400
  name: Insert Items
  is_ignore: false
- id: 500
  name: Close Box
  is_ignore: false
- id: 600
  name: Attach Box Label
  is_ignore: false
- id: 700
  name: Scan Label
  is_ignore: false
- id: 800
  name: Attach Shipping Label
  is_ignore: false
- id: 900
  name: Put on Back Table
  is_ignore: false
- id: 1000
  name: Fill out Order
  is_ignore: false
- id: 8100
  name: 'Null'
```

```
  is_ignore: true
```

Visualizamos los metadatos de la etiqueta almacenados en: optk.configs.datasets.annotations.ACTIVITY_1S_ANNOTATION, que por defecto trae valores genéricos, si cargamos la etiqueta de la sesión S0100 del usuario definido previamente se completarán adaptándose a ese escenario.

[4]:
```
print("conf_type:", optk.configs.datasets.annotations.ACTIVITY_1S_ANNOTATION.
  ↪conf_type)
print("name:", optk.configs.datasets.annotations.ACTIVITY_1S_ANNOTATION.name)
print(OmegaConf.to_yaml(optk.configs.datasets.annotations.
  ↪ACTIVITY_1S_ANNOTATION.path))
```

```
conf_type: ANNOT_FILE
name: activity-1s
dir: ${path.openpack.rootdir}/${user.name}/annotation/activity-1s/
fname: ${session}.csv
```

Fijando en nuestra configuración los distintos metadatos que se encuentran en el toolkit prefijados nos facilitará la navegación por los directorios del dataset.

[5]:
```
cfg.dataset.annotation = optk.configs.datasets.annotations.
  ↪ACTIVITY_1S_ANNOTATION
cfg.session = "S0100"

path = Path(
    cfg.dataset.annotation.path.dir,
    cfg.dataset.annotation.path.fname,
)
print(path)
```

```
data\datasets\openpack\v0.3.1\U0101\annotation\activity-1s\S0100.csv
```

Cargando y visualizando el contenido vemos el instante en timestamp (ms) con una etiqueta por segundo , usuario, sesión, nº de caja en curso, etiqueta de operación y etiqueta de acción.

[6]:
```
df_labels = pd.read_csv(path)
df_labels.head(15)
```

[6]:

	unixtime	user	session	box	operation	action
0	1634178335000	U0101	S0100	1	100	101
1	1634178336000	U0101	S0100	1	100	101
2	1634178337000	U0101	S0100	1	100	101
3	1634178338000	U0101	S0100	1	100	101
4	1634178339000	U0101	S0100	1	100	8100
5	1634178340000	U0101	S0100	1	100	8100
6	1634178341000	U0101	S0100	1	100	8100

7	1634178342000	U0101	S0100	1	100	108
8	1634178343000	U0101	S0100	1	100	8100
9	1634178344000	U0101	S0100	1	100	8100
10	1634178345000	U0101	S0100	1	100	8100
11	1634178346000	U0101	S0100	1	100	8100
12	1634178347000	U0101	S0100	1	100	8100
13	1634178348000	U0101	S0100	1	200	204
14	1634178349000	U0101	S0100	1	200	204

El Toolkit nos proporciona una herramienta muy últil para realizar emparejamientos de tiempos entre los distintos sensores. Las etiquetas están asignadas a cada segundo, pero cada sensor tiene un periódo de refresco, p.e. los keypoints tienen una frecuencia de 15 por segundo, los sensores IMU tienen 30, etc. Hacemos una prueba con etiquetqas que cambian entre dos segundos con una frecuencia de muestreo > 1Hz.

[7]:
```
unixtimes = np.array([1634178347000,
                      1634178347200,
                      1634178347500,
                      1634178347900,
                      1634178348100,
                      ])
```

Esta función nos devuelve la carga de la etiqueta asociando a su vez para cada instante del nuevo muestreo los datos contenidos en la etiqueta original, siendo útil para unificar los resultados obtenidos para cada sensor una vez contemos con modelos entrenados para cada uno.

[8]:
```
df_annot_resampled = optk.data.load_and_resample_operation_labels(path,␣
 ↪unixtimes, classes=optk.OPENPACK_OPERATIONS)
df_annot_resampled
```

[8]:

	unixtime	annot_time	user	session	box	act_id	act_idx
0	1634178347000	1634178347000	U0101	S0100	1	100	0
1	1634178347200	1634178347000	U0101	S0100	1	100	0
2	1634178347500	1634178347000	U0101	S0100	1	100	0
3	1634178347900	1634178347000	U0101	S0100	1	100	0
4	1634178348100	1634178348000	U0101	S0100	1	200	1

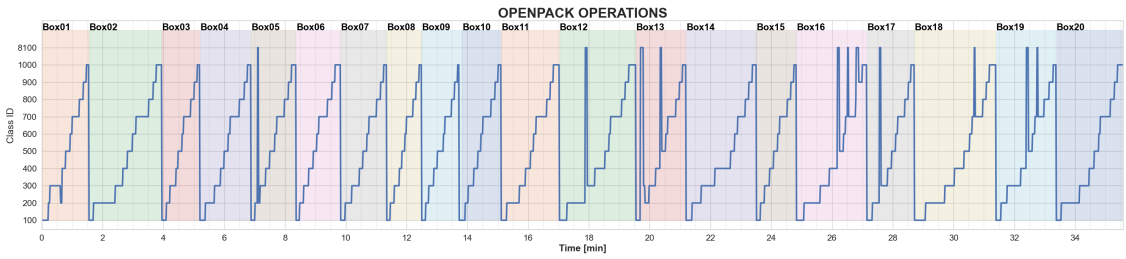
A continuación visualizaremos el flujo de operaciones de una sesión completa, para ello nos ayudaremos de la siguiente función que imprimirá la línea de tiempo desglosando operación en vertical, y caja en bloques de colores utilizando una imagen. Las funciones de visualización han sido extraídas de la documentación oficial de Openpack Toolkit

[9]:
```
@noglobal()
def plot_openpack_operations(df: pd.DataFrame, xlim=None, figsize=(30, 7),␣
 ↪OPENPACK_OPERATIONS=optk.OPENPACK_OPERATIONS):
    seq_len = len(df)
```

```
    df["cls_idx"] = optk.OPENPACK_OPERATIONS.
↪convert_id_to_index(df["operation"])

    df_head = df.drop_duplicates(["user", "session", "box"], keep="first")
    df_tail = df.drop_duplicates(["user", "session", "box"], keep="last")
    df_box = pd.DataFrame({
        "box": df_head["box"],
        "start": df_head.index,
        "end": df_tail.index,
    }).reset_index(drop=True)

    # == Plot ==
    fig, ax0 = plt.subplots(1, 1, figsize=figsize)
    xloc = np.arange(seq_len)

    ax0.plot(xloc, df["cls_idx"], lw=3)
    for index, row in df_box.iterrows():
        ax0.fill_between([row.start, row.end], 0, 11, color=f"C{row.box%10}",␣
↪alpha=0.2)
        ax0.text(
            row.start, 11, f"Box{row.box:0=2}",
            fontweight="bold", color="black",
        )

    xticks = np.arange(0, seq_len, 60 * 2)
    xticks_minor = np.arange(0, seq_len, 30)
    ax0.set_xticks(xticks)
    ax0.set_xticklabels(xticks // 60)
    ax0.set_xticks(xticks_minor, minor=True)
    ax0.set_xlabel("Time [min]", fontweight="bold")
    if xlim is None:
        ax0.set_xlim([0, seq_len])
    else:
        ax0.set_xlim(xlim)

    yticklabels = [k for k in OPENPACK_OPERATIONS.get_ids()]
    ax0.set_yticks(np.arange(len(OPENPACK_OPERATIONS)))
    ax0.set_yticklabels(yticklabels)
    ax0.set_ylabel("Class ID")

    ax0.grid(True, which="minor", linestyle=":")

    ax0.set_title(f"OPENPACK OPERATIONS", fontsize="x-large", fontweight="bold")

    fig.tight_layout()
    return fig
```

[10]:
```
fig = plot_openpack_operations(df_labels)
fig.show()
```

```
C:\Users\javier.delafuente\AppData\Local\Temp\ipykernel_14960\766994650.py:2:
UserWarning: Matplotlib is currently using
module://matplotlib_inline.backend_inline, which is a non-GUI backend, so cannot
show the figure.
  fig.show()
```



1.2 IMUs

Una vez vistas las etiquetas podemos ver también el contenido de los datos de sensores IMU, de los que se cuenta con 4 ATR-TSND151, dos en cada brazo. Se cuenta con datos de giroscopio, acelerómetro y cuaternión. Podemos ver los metadatos de este dispositivo a continuación:

[11]:
```
(OmegaConf.to_yaml(optk.configs.datasets.streams.ATR_QAGS_STREAM))
```

[11]:
```
'schema: ImuConfig\nname: atr-qags\ndescription: null\nsuper_stream:
None\npath:\n  dir: ${path.openpack.rootdir}/${user.name}/atr/${device}\n
fname: ${session}.csv\nfile_format: null\nframe_rate: 30\ndevices:\n- atr01\n-
atr02\n- atr03\n- atr04\nacc: true\ngyro: true\nquat: true\n'
```

[12]:
```
print(OmegaConf.to_yaml(cfg))
```

```
user:
  id: 101
  name: U0101
  sessions:
    S0100:
      duration: 35m36s
      end: '2021-10-14T12:01:10+09:00'
      start: '2021-10-14T11:25:34+09:00'
    S0200:
      duration: 28m54s
      end: '2021-10-14T13:46:36+09:00'
      start: '2021-10-14T13:17:42+09:00'
    S0300:
      duration: 28m58s
```

```
      end: '2021-10-14T14:38:08+09:00'
      start: '2021-10-14T14:09:10+09:00'
    S0400:
      duration: 30m55s
      end: '2021-10-14T15:30:43+09:00'
      start: '2021-10-14T14:59:48+09:00'
    S0500:
      duration: 27m35s
      end: '2021-10-14T16:21:40+09:00'
      start: '2021-10-14T15:54:05+09:00'
session: S0100
path:
  openpack:
    version: v0.3.1
    rootdir: data/datasets/openpack/v0.3.1
dataset:
  annotation:
    conf_type: ANNOT_FILE
    name: activity-1s
    version: ''
    path:
      dir: ${path.openpack.rootdir}/${user.name}/annotation/activity-1s/
      fname: ${session}.csv
    file_format: null
    classes: ???
    activity_sets:
      openpack-operations:
        classes:
        - id: 100
          name: Picking
          is_ignore: false
        - id: 200
          name: Relocate Item Label
          is_ignore: false
        - id: 300
          name: Assemble Box
          is_ignore: false
        - id: 400
          name: Insert Items
          is_ignore: false
        - id: 500
          name: Close Box
          is_ignore: false
        - id: 600
          name: Attach Box Label
          is_ignore: false
        - id: 700
          name: Scan Label
```

```
    is_ignore: false
  - id: 800
    name: Attach Shipping Label
    is_ignore: false
  - id: 900
    name: Put on Back Table
    is_ignore: false
  - id: 1000
    name: Fill out Order
    is_ignore: false
  - id: 8100
    name: 'Null'
    is_ignore: true
openpack-actions:
  classes:
  - id: 101
    name: Pick Up Sheet
    is_ignore: false
  - id: 103
    name: Pick Up Item from Box
    is_ignore: false
  - id: 107
    name: Pick Up Box Sheet
    is_ignore: false
  - id: 108
    name: Walk to Work Bench
    is_ignore: false
  - id: 109
    name: Put Packed Box
    is_ignore: false
  - id: 201
    name: Remove Item Label
    is_ignore: false
  - id: 202
    name: Attach to Order Sheet
    is_ignore: false
  - id: 203
    name: Hold Pen
    is_ignore: false
  - id: 204
    name: Write Check Mark
    is_ignore: false
  - id: 205
    name: Put Item Small Bag
    is_ignore: false
  - id: 301
    name: Pick Cardboard
    is_ignore: false
```

```
- id: 302
  name: Bend Flap
  is_ignore: false
- id: 303
  name: Attach Tape
  is_ignore: false
- id: 304
  name: Turn Over Box
  is_ignore: false
- id: 305
  name: Pick Up Assembled Box
  is_ignore: false
- id: 401
  name: Insert Item into Box
  is_ignore: false
- id: 402
  name: Air Cushion
  is_ignore: false
- id: 403
  name: Separate Air Cushion
  is_ignore: false
- id: 404
  name: Put Item Small Bag
  is_ignore: false
- id: 501
  name: Bend Flap
  is_ignore: false
- id: 502
  name: Attach Tape
  is_ignore: false
- id: 601
  name: Attach Box Label
  is_ignore: false
- id: 701
  name: Pick Up HT
  is_ignore: false
- id: 702
  name: Scan Order Sheet
  is_ignore: false
- id: 703
  name: Scan Box
  is_ignore: false
- id: 704
  name: Scan Item
  is_ignore: false
- id: 705
  name: Hold Scanner
  is_ignore: false
```

```
      - id: 706
        name: Scan Order Sheet
        is_ignore: false
      - id: 707
        name: Scan Printer
        is_ignore: false
      - id: 801
        name: Pick Up Shipping Label
        is_ignore: false
      - id: 802
        name: Attach Shipping Label
        is_ignore: false
      - id: 901
        name: Pick Up Packed Box
        is_ignore: false
      - id: 902
        name: Put Packed Box
        is_ignore: false
      - id: 1001
        name: Pick Up Pen
        is_ignore: false
      - id: 1002
        name: Write Sign
        is_ignore: false
      - id: 1003
        name: Push Order Sheet into Tray
        is_ignore: false
      - id: 8101
        name: Others
        is_ignore: true
      - id: 8102
        name: System Error
        is_ignore: true
      - id: 8103
        name: Ignore
        is_ignore: true
      - id: 8104
        name: Unknown
        is_ignore: true
      - id: 8201
        name: System Error
        is_ignore: true
  stream: null
```

Similar a la carga de las etiquetas, de manera sencilla añadimos los metadatos al stream de la configuración y cargamos los de la IMU 01 que se corresponde con la muñeca de la mano derecha.

[13]:
```
cfg.dataset.stream = optk.configs.datasets.streams.ATR_QAGS_STREAM
cfg.device='atr01'
path = Path(cfg.dataset.stream.path.dir, cfg.dataset.stream.path.fname)
df_attr_1 = pd.read_csv(path)
df_attr_1.head()
```

[13]:

	unixtime	acc_x	acc_y	acc_z	gyro_x	gyro_y	gyro_z	quat_w	\
0	1634178333023	0.2838	0.9830	0.0572	-3.21	127.65	7.39	0.0	
1	1634178333056	0.2513	0.9357	0.0733	-6.23	155.37	11.12	0.0	
2	1634178333089	0.1649	0.9249	0.0713	-6.70	150.33	15.38	0.0	
3	1634178333122	0.1222	0.9801	0.1050	-2.93	108.87	20.03	0.0	
4	1634178333155	0.1551	1.0636	0.1053	-0.92	77.08	22.49	0.0	

	quat_x	quat_y	quat_z
0	0.0	0.0	0.0
1	0.0	0.0	0.0
2	0.0	0.0	0.0
3	0.0	0.0	0.0
4	0.0	0.0	0.0

A continuación visualizamos los datos contenidos de manera similar a las etiquetas separando en visualizaciones los datos de giroscopio, acelerómetro y cuaternión.

[14]:
```
@noglobal()
def plot_atr_qags(df: pd.DataFrame, cfg: DictConfig):
    seq_len = len(df)


    fig = plt.figure(figsize=(30, 2.5 * 4))
    gs_master = GridSpec(nrows=2, ncols=1, height_ratios=[3, 1])
    gs_line = GridSpecFromSubplotSpec(
        nrows=3, ncols=1, subplot_spec=gs_master[0], hspace=0.05)
    gs_hist = GridSpecFromSubplotSpec(nrows=1, ncols=3,␣
 ↪subplot_spec=gs_master[1])

    data = [
        {
            "label": "Acc [G]",
            "cols": ["acc_x", "acc_y", "acc_z"],
            "lim": [-4.0, 4.0],
        },
        {
            "label": "Gyro [dps]",
            "cols": ["gyro_x", "gyro_y", "gyro_z"],
            "lim": [-500.0, 500.0],
        },
        {
```

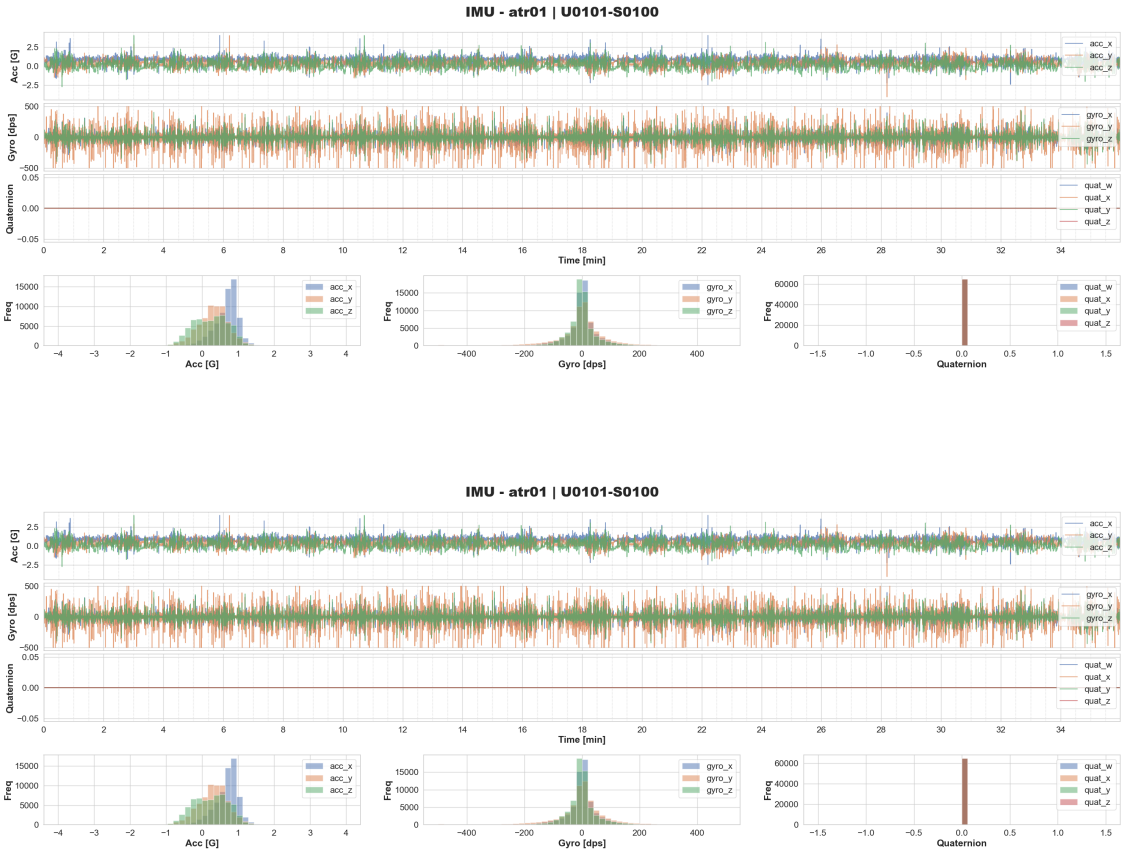
```
            "label": "Quaternion",
            "cols": ["quat_w", "quat_x", "quat_y", "quat_z"],
            "lim": [-1.5, 1.5],
        },
    ]
    xloc = df.index.values
    for i, d in enumerate(data):
        cols = d["cols"]
        ylabel = d["label"]
        lim = d["lim"]

        X = df[cols].values.T

        # -- Sequence (Acc / Gyro / Quat) --
        ax0 = fig.add_subplot(gs_line[i])
        for ch, col_name in enumerate(cols):
            ax0.plot(xloc, X[ch], label=col_name, color=f"C{ch}", alpha=0.75)

        xticks = np.arange(0, seq_len + 1, 30 * 60 * 2)
        xticks_minor = np.arange(0, seq_len + 1, 30 * 30)
        ax0.set_xticks(xticks)
        ax0.set_xticklabels(xticks // (30 * 60))
        ax0.set_xticks(xticks_minor, minor=True)
        ax0.set_xlim([0, seq_len])

        ax0.set_ylabel(ylabel, fontweight="bold")

        ax0.grid(True, which="minor", linestyle=":")
        ax0.legend(loc="upper right")

        if i == 2:
            ax0.set_xlabel("Time [min]", fontweight="bold")
        else:
            ax0.tick_params(
                labelbottom=False
            )

        # -- Histgram --
        ax1 = fig.add_subplot(gs_hist[i])
        for ch, col_name in enumerate(cols):
            ax1.hist(
                X[ch],
                range=lim,
                bins=50,
                label=col_name,
                color=f"C{ch}",
                alpha=0.50)
```

```
        ax1.set_xlabel(ylabel, fontweight="bold")
        ax1.set_ylabel("Freq", fontweight="bold")
        ax1.legend(loc="upper right")

    fig.suptitle(
        f"IMU - {cfg.device} | {cfg.user.name}-{cfg.session}",
        fontsize="x-large",
        fontweight="black")
    fig.tight_layout()
    return fig
```

Vemos que los datos de cuaternión no han sido recogidos, por lo tanto se descartarán.

[15]:
```
plot_atr_qags(df_attr_1, cfg)
```

[15]:



1.3 Sensores E4

Se cuenta con dos sensores Empatica 4 que registrarán también los movimientos de acelerómetro además de presión arterial, actividad electrodérmica y temperatura. Las cargas de cada dato se realizarán por separado y las iremos viendo a continuación.

1.3.1 Acelerómetro

[16]:
```
print(OmegaConf.to_yaml(optk.configs.datasets.streams.E4_ACC_STREAM))
```

```
schema: ImuConfig
name: e4-acc
description: null
super_stream: null
path:
  dir: ${path.openpack.rootdir}/${user.name}/e4/${device}/acc
  fname: ${session}.csv
file_format: null
frame_rate: 32
devices:
- e401
- e402
sensor: ''
```

[17]:
```
cfg.device='e401'
cfg.dataset.stream=optk.configs.datasets.streams.E4_ACC_STREAM
path = Path(cfg.dataset.stream.path.dir, cfg.dataset.stream.path.fname)
df_e4_acc_1 = pd.read_csv(path)
df_e4_acc_1.head()
```

[17]:

	time	acc_x	acc_y	acc_z
0	1634178334012	-0.030273	-0.094005	-0.006373
1	1634178334043	-0.031866	-0.101972	-0.006373
2	1634178334075	-0.027086	-0.105158	-0.004780
3	1634178334106	-0.015933	-0.103565	0.007967
4	1634178334137	-0.011153	-0.101972	0.014340

1.3.2 Presión Sanguínea (BVP)

[18]:
```
cfg.device='e401'
cfg.dataset.stream=optk.configs.datasets.streams.E4_BVP_STREAM
path = Path(cfg.dataset.stream.path.dir, cfg.dataset.stream.path.fname)
df_e4_bvp_1 = pd.read_csv(path)
df_e4_bvp_1.head()
```

[18]:

	time	bvp
0	1634178334010	-115.49
1	1634178334026	-125.91
2	1634178334041	-130.49
3	1634178334057	-128.92
4	1634178334073	-121.89

1.3.3 Actividad Electrodérmica (EDA)

[19]:
```
cfg.device='e401'
cfg.dataset.stream=optk.configs.datasets.streams.E4_EDA_STREAM
path = Path(cfg.dataset.stream.path.dir, cfg.dataset.stream.path.fname)
df_e4_eda_1 = pd.read_csv(path)
df_e4_eda_1.head()
```

[19]:

	time	eda
0	1634178334229	0.198885
1	1634178334479	0.205292
2	1634178334729	0.204011
3	1634178334979	0.193760
4	1634178335229	0.195041

1.3.4 Temperatura

[20]:
```
cfg.device='e401'
cfg.dataset.stream=optk.configs.datasets.streams.E4_TEMP_STREAM
path = Path(cfg.dataset.stream.path.dir, cfg.dataset.stream.path.fname)
df_e4_temp_1 = pd.read_csv(path)
df_e4_temp_1.head()
```

[20]:

	time	temp
0	1634178334229	33.65
1	1634178334479	33.71
2	1634178334729	33.71
3	1634178334979	33.71
4	1634178335229	33.71

Englobamos todo en una única visualización para obtener un contexto más general.

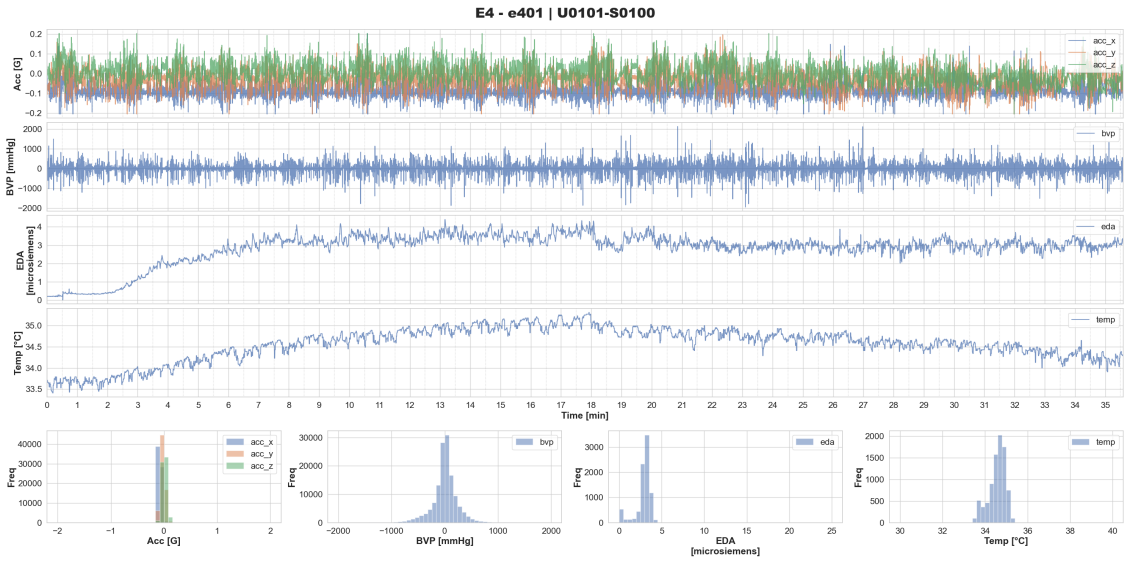
[21]:
```
@noglobal()
def plot_e4_all(
        data: dict,
        user: str,
        session: str,
        device: str,
        version=None,
        xlim=None,
        figsize=(30, 7)):

    E4_SENSORS = ["acc", "bvp", "eda", "temp"]

    # == Plot ==
    fig = plt.figure(figsize=(30, 3 * 5))
    gs_master = GridSpec(nrows=2, ncols=1, height_ratios=[4, 1])
```

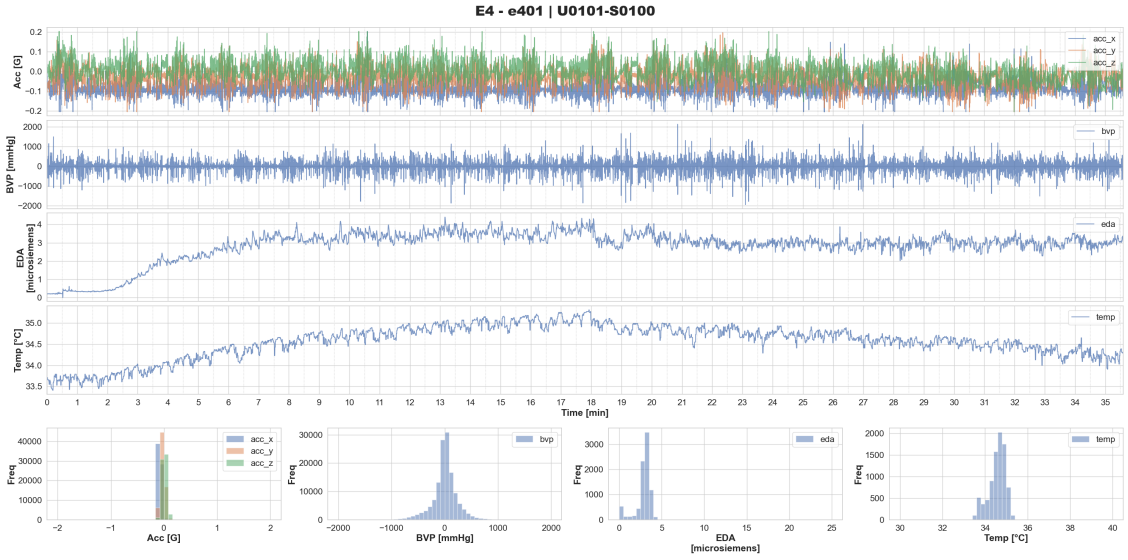
```
    gs_line = GridSpecFromSubplotSpec(
        nrows=4, ncols=1, subplot_spec=gs_master[0], hspace=0.05)
    gs_hist = GridSpecFromSubplotSpec(nrows=1, ncols=4,␣
↪subplot_spec=gs_master[1])

    metadata = {
        "acc": {
            "label": "Acc [G]",
            "cols": ["acc_x", "acc_y", "acc_z"],
            "lim": [-2.0, 2.0],
            "fs": 32,
        },
        "bvp": {
            "label": "BVP [mmHg]",
            "cols": ["bvp"],
            "lim": [-2000.0, 2000.0],
            "fs": 64,
        },
        "eda": {
            "label": "EDA\n[microsiemens]",
            "cols": ["eda"],
            "lim": [0.0, 25.0],
            "fs": 4,
        },
        "temp": {
            "label": "Temp [°C]",
            "cols": ["temp"],
            "lim": [30.0, 40.0],
            "fs": 4,
        },
    }
    # xloc = df.index.values
    for i, sensor in enumerate(E4_SENSORS):
        d = metadata[sensor]

        df = data[sensor]
        cols = d["cols"]
        ylabel = d["label"]
        lim = d["lim"]
        fs = d["fs"]

        X = df[cols].values.T
        xloc = df.index.values
        seq_len = len(xloc)

        # -- Sequence (Acc / Gyro / Quat) --
        ax0 = fig.add_subplot(gs_line[i])
```

```
        for ch, col_name in enumerate(cols):
            ax0.plot(xloc, X[ch], label=col_name, color=f"C{ch}", alpha=0.75)

        xticks = np.arange(0, seq_len + 1, fs * 60)
        xticks_minor = np.arange(0, seq_len + 1, fs * 30)
        ax0.set_xticks(xticks)
        ax0.set_xticklabels(xticks // (fs * 60))
        ax0.set_xticks(xticks_minor, minor=True)
        ax0.set_xlim([0, seq_len])

        ax0.set_ylabel(ylabel, fontweight="bold")
        ax0.grid(True, which="minor", linestyle=":")
        ax0.legend(loc="upper right")

        if i == 3:
            ax0.set_xlabel("Time [min]", fontweight="bold")
        else:
            ax0.tick_params(
                labelbottom=False
            )

        # -- Histgram --
        ax1 = fig.add_subplot(gs_hist[i])
        for ch, col_name in enumerate(cols):
            ax1.hist(
                X[ch],
                range=lim,
                bins=50,
                label=col_name,
                color=f"C{ch}",
                alpha=0.50)

        ax1.set_xlabel(ylabel, fontweight="bold")
        ax1.set_ylabel("Freq", fontweight="bold")
        ax1.legend(loc="upper right")

    fig.suptitle(
        f"E4 - {device} | {user}-{session}",
        fontsize="x-large",
        fontweight="black")
    fig.tight_layout()
    return fig
```

[22]:
```
plot_e4_all( data={"acc": df_e4_acc_1, "bvp": df_e4_bvp_1, "eda": df_e4_eda_1,␣
↪"temp": df_e4_temp_1}, user =cfg.user.name, session = cfg.session, device =␣
↪cfg.device)
```

[22]:





2 Kinect

Visualizamos el contenido relativo a los *keypoints* obtenidos del sensor Kinect utilizando open-mmlab/mmpose. En primer lugar, estos son los metadatos del toolkit.

[23]:
```
print(OmegaConf.to_yaml(optk.configs.datasets.streams.KINECT_2D_KPT_STREAM))
```

```
schema: KeypointConfig
name: kinect-2d-kpt
description: null
```

```
super_stream: None
path:
  dir:
${path.openpack.rootdir}/${user.name}/kinect/${..category}/${..model}/single
  fname: ${session}.json
file_format: null
frame_rate: 15
category: 2d-kpt
model: mmpose-hrnet-w48-posetrack18-384x288-posewarper-stage2
nodes:
  0: nose
  1: left_eye
  2: right_eye
  3: left_ear
  4: right_ear
  5: left_shoulder
  6: right_shoulder
  7: left_elbow
  8: right_elbow
  9: left_wrist
  10: right_wrist
  11: left_hip
  12: right_hip
  13: left_knee
  14: right_knee
  15: left_ankle
  16: right_ankle
```

[24]:
```
cfg.dataset.stream=optk.configs.datasets.streams.KINECT_2D_KPT_STREAM
path = Path(cfg.dataset.stream.path.dir, cfg.dataset.stream.path.fname)
with open(path, "r") as f:
    data = json.load(f)
```

Una vez cargado el json tenemos un diccionario con la información general, licencias, categorías que indican cada punto y sus relaciones con el resto y, finalmente, las anotaciones tanto en keypoint como en caja.

[25]:
```
data["info"]
```

[25]:
```
{'year': 2022,
 'version': 'v0.2.0',
 'description': 'openpack/U0101/2d-kpt/mmpose-
hrnet-w48-posetrack18-384x288-posewarper-stage2/single/S0100',
 'contributor': 'Naoya Yoshimura, Jaime Morales, Takuya Maekawa',
 'url': 'https://open-pack.github.io',
 'date_created': '2022/03/31'}
```

[26]:
```
data["licenses"]
```

[26]:
```
[{'id': 0,
  'name': 'Creative Commons Attribution Non Commercial Share Alike 4.0
International',
  'url': 'https://open-pack.github.io'}]
```

[27]:
```
data["categories"]
```

[27]:
```
[{'supercategory': 'person',
  'id': 1,
  'name': 'person',
  'keypoints': ['nose',
   'left_eye',
   'right_eye',
   'left_ear',
   'right_ear',
   'left_shoulder',
   'right_shoulder',
   'left_elbow',
   'right_elbow',
   'left_wrist',
   'right_wrist',
   'left_hip',
   'right_hip',
   'left_knee',
   'right_knee',
   'left_ankle',
   'right_ankle'],
  'skeleton': [[16, 14],
   [14, 12],
   [17, 15],
   [15, 13],
   [12, 13],
   [6, 12],
   [7, 13],
   [6, 7],
   [6, 8],
   [7, 9],
   [8, 10],
   [9, 11],
   [2, 3],
   [1, 2],
   [1, 3],
   [2, 4],
   [3, 5],
   [4, 6],
```

```
    [5, 7]]}]
```

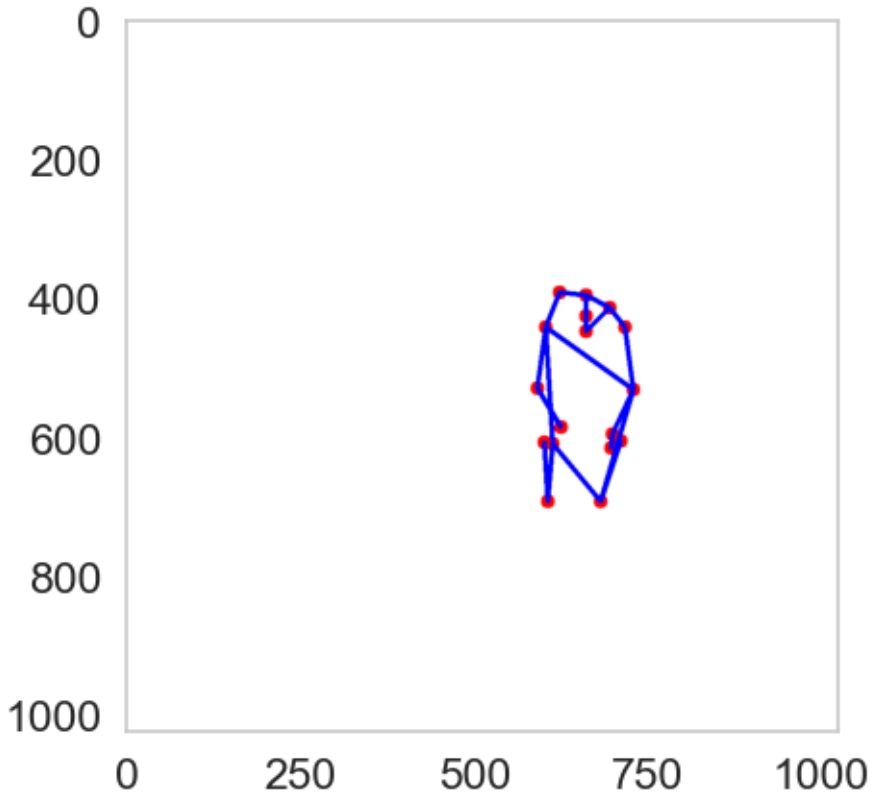
[28]:
```
data["annotations"][0]
```

[28]:
```
{'id': 1634178332050000,
 'image_id': 1634178332050,
 'category_id': 1,
 'area': 34339.07,
 'bbox': [595.1, 331.8, 110.3, 311.3],
 'iscrowd': 0,
 'keypoints': [[627.3, 385.3, 0.9519461],
  [655.8, 407.6, 0.8386567],
  [631.4, 356.8, 0.5640739],
  [672.1, 385.3, 0.40338063],
  [649.7, 362.9, 0.28661534],
  [672.1, 425.9, 0.8711105],
  [643.6, 423.9, 0.73448473],
  [676.1, 521.5, 0.9056189],
  [641.6, 497.1, 0.52730274],
  [655.8, 606.9, 0.7874441],
  [647.7, 570.3, 0.13315539],
  [657.8, 588.6, 0.62024885],
  [627.3, 580.5, 0.54534656],
  [655.8, 674.0, 0.44440606],
  [633.4, 661.8, 0.42726257],
  [668.0, 627.2, 0.2758708],
  [635.5, 651.6, 0.2701233]],
 'num_keypoints': 17,
 'bbox_score': 0.98887616,
 'track_id': 9}
```

Visualizamos un fotograma extraído utilizando OpenCV.

[29]:
```
canvas_size = (1024, 1024, 3)
canvas = np.ones(canvas_size)
def draw_points(canvas, keypoints):
    for point in keypoints:
        x, y = int(np.round(point[0])), int(np.round(point[1]))
        #print(x, y)
        canvas = cv2.circle(canvas, (x, y), radius=0, color=(1, 0, 0),␣
 ↪thickness=20)
    return canvas
def draw_bones(canvas, pairs, points):
    for pair in pairs:
        try:
            canvas = draw_edge(canvas, points[pair[0]][:2], points[pair[1]][:2])
        except:
```

```
            pass
    return canvas
def draw_edge(canvas, coord1, coord2):
    x1, y1 = int(np.round(coord1[0])), int(np.round(coord1[1]))
    x2, y2 = int(np.round(coord2[0])), int(np.round(coord2[1]))
    canvas = cv2.line(canvas, (x1, y1), (x2, y2), color=(0, 0, 1), thickness=5)
    return canvas
canvas = draw_points(canvas, data["annotations"][1000]["keypoints"])
canvas = draw_bones(canvas, data["categories"][0]["skeleton"],␣
 ↪data["annotations"][1000]["keypoints"])
plt.imshow(canvas)
plt.grid(False)
plt.show()
```



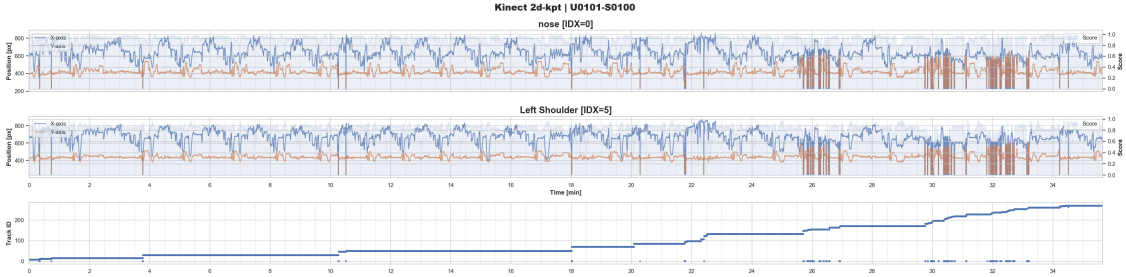
Y al igual que en los anteriores casos también visualizamos los movimientos de dos puntos en los ejes vertical y horizontal además del track id

[30]:
```
@noglobal()
def plot_kinect_2d_kpt(annots: dict, cfg: DictConfig):
    FS = 15
    seq_len = len(annots)
```

```
    # == Plot ==
    sns.set("notebook", "whitegrid")
    fig = plt.figure(figsize=(30, 2.5 * 3))
    gs_master = GridSpec(nrows=3, ncols=1)

    # -- Keypoints Location --
    data = [
        {
            "node": "nose",
            "idx": 0,
        },
        {
            "node": "Left Shoulder",
            "idx": 5,
        },
    ]
    xloc = np.arange(len(annots))
    for i, d in enumerate(data):
        title = d["node"]
        kpt_idx = d["idx"]

        X = np.array([annots[i]["keypoints"] for i in range(len(annots))])

        # -- Sequence (Acc / Gyro / Quat) --
        ax0 = fig.add_subplot(gs_master[i])
        ax1 = ax0.twinx()

        # prediction score
        ax1.fill_between(xloc, X[:, kpt_idx, 2], label="Score", color="C0",␣
↪alpha=0.1)
        # Plot position
        for ch in range(2):
            ax0.plot(
                xloc,
                X[:, kpt_idx, ch],
                label="X-axis" if ch == 0 else "Y-axis",
                color=f"C{ch}",
                alpha=0.75,
            )

        xticks = np.arange(0, seq_len + 1, FS * 60 * 2)
        xticks_minor = np.arange(0, seq_len + 1, FS * 30)
        ax0.set_xticks(xticks)
        ax0.set_xticklabels(xticks // (FS * 60))
        ax0.set_xticks(xticks_minor, minor=True)
        ax0.set_xlim([0, seq_len])
```

```
        ax0.set_ylabel("Position [px]", fontweight="bold")

        ax1.set_yticks(np.arange(0, 1.1, 0.2))
        ax1.set_ylabel("Score", fontweight="bold")

        ax0.grid(True, which="minor", linestyle=":")
        ax0.legend(loc="upper left")
        ax1.legend(loc="upper right")
        ax0.set_title(f"{title} [IDX={kpt_idx}]", fontweight="bold",␣
↪fontsize="x-large")

        if i == 1:
            ax0.set_xlabel("Time [min]", fontweight="bold")
        else:
            ax0.tick_params(
                labelbottom=False
            )

    # -- [2] Tracking Id --
    ax0 = fig.add_subplot(gs_master[2])
    X = np.array([annots[i]["track_id"] for i in range(len(annots))])
    ax0.scatter(xloc, X, color="C0", alpha=0.75, s=5)
    ax0.set_xticks(xticks)
    ax0.set_xticklabels(xticks // (FS * 60))
    ax0.set_xticks(xticks_minor, minor=True)
    ax0.set_xlim([0, seq_len])
    ax0.set_ylabel("Track ID", fontweight="bold")
    ax0.grid(True, which="minor", linestyle=":")

    fig.suptitle(
        f"Kinect 2d-kpt | {cfg.user.name}-{cfg.session}",
        fontsize="x-large",
        fontweight="black")
    fig.tight_layout()
    return fig,
```

[31]:
```
fig = plot_kinect_2d_kpt(data["annotations"], cfg)
```



2.1 Señales de sistema

El resto de datos pertenecen a señales proporcionadas tanto por dispositivos auxiliares del operador como por las características de las operaciones.

2.1.1 Lector de etiquetas

Los datos proporcionados por el lector de etiquetas al escanear un pedido quedan almacenados aquí.

[76]:
```
print(OmegaConf.to_yaml(optk.configs.datasets.streams.
  ↪SYSTEM_HT_ORIGINAL_STREAM))
```

```
schema: SystemDataConfig
name: system-ht-original
description: null
super_stream: None
path:
  dir: ${path.openpack.rootdir}/${user.name}/system/ht
  fname: ${session}.csv
file_format: null
frame_rate: -1
devices: ???
acc: true
gyro: true
quat: true
```

[78]:
```
cfg.dataset.stream = optk.configs.datasets.streams.SYSTEM_HT_ORIGINAL_STREAM
path = Path(cfg.dataset.stream.path.dir, cfg.dataset.stream.path.fname)
df_ht = pd.read_csv(path)
df_ht.head()
```

[78]:

	unixtime	datetime	order_sheet	box	item
0	1634178394000	2021-10-14 11:26:34+09:00	START	-----	DEN010000
1	1634178395000	2021-10-14 11:26:35+09:00	START	-----	BOX01
2	1634178397000	2021-10-14 11:26:37+09:00	START	-----	ITEM304
3	1634178519000	2021-10-14 11:28:39+09:00	START	-----	DEN010001
4	1634178521000	2021-10-14 11:28:41+09:00	START	-----	BOX03

2.1.2 Listado de pedidos

A modo de información también se recoge el listado de pedidos con los objetos de cada caja y sus cantidades.

[40]:
```
cfg.dataset.stream = optk.configs.datasets.streams.SYSTEM_ORDER_SHEET_STREAM
path = Path(cfg.dataset.stream.path.dir, cfg.dataset.stream.path.fname)
```

```
df_order = pd.read_csv(path)
df_order.head()
```

[40]:

	Unnamed: 0	sheet_no	session	box	pattern	total_amount	item1	item2	item3	item4	item5	amount1	amount2	amount3	amount4	amount5
0	0	DEN010000	S0100	0	xM1	1	304	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
1	1	DEN010001	S0100	1	xS4-5	4	210	212.0	217.0	222.0	NaN	1	1.0	1.0	1.0	NaN
2	2	DEN010002	S0100	2	xS1	1	106	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
3	3	DEN010003	S0100	3	xS2-3	2	110	214.0	NaN	NaN	NaN	1	1.0	NaN	NaN	NaN
4	4	DEN010004	S0100	4	xS1	1	203	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN

[42]:
```
df_order
```

[42]:

	Unnamed: 0	sheet_no	session	box	pattern	total_amount	item1	item2	item3	item4	item5	amount1	amount2	amount3	amount4	amount5
0	0	DEN010000	S0100	0	xM1	1	304	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
1	1	DEN010001	S0100	1	xS4-5	4	210	212.0	217.0	222.0	NaN	1	1.0	1.0	1.0	NaN
2	2	DEN010002	S0100	2	xS1	1	106	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
3	3	DEN010003	S0100	3	xS2-3	2	110	214.0	NaN	NaN	NaN	1	1.0	NaN	NaN	NaN
4	4	DEN010004	S0100	4	xS1	1	203	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
5	5	DEN010005	S0100	5	xS1	1	109	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
6	6	DEN010006	S0100	6	xL1	1	403	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
7	7	DEN010007	S0100	7	xS1	1	107	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
8	8	DEN010008	S0100	8	xS1	1	220	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
9	9	DEN010009	S0100	9	xS1	1	219	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
10	10	DEN010010	S0100	10	xS4-5	4	103	201.0	NaN	NaN	NaN	1	1.0	NaN	NaN	NaN
11	11	DEN010011	S0100	11	xS4-5	5	102	108.0	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
12	12	DEN010012	S0100	12	xM1	1	308	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
13	13	DEN010013	S0100	13	xS2-3	2	305	311.0	NaN	NaN	NaN	1	1.0	NaN	NaN	NaN
14	14	DEN010014	S0100	14	xS1	1	104	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
15	15	DEN010015	S0100	15	xM2-5	3	401	402.0	NaN	NaN	NaN	1	1.0	NaN	NaN	NaN
16	16	DEN010016	S0100	16	xS2-3	2	213	223.0	NaN	NaN	NaN	1	1.0	NaN	NaN	NaN
17	17	DEN010017	S0100	17	xM2-5	4	306	309.0	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
18	18	DEN010018	S0100	18	xL1	1	501	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
19	19	DEN010019	S0100	19	xM2-5	2	301	307.0	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN

5	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
7	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
8	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
9	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
10	204.0	216.0	NaN	1	1.0	1.0	1.0	NaN
11	206.0	208.0	225.0	1	1.0	1.0	1.0	1.0
12	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
13	NaN	NaN	NaN	1	1.0	NaN	NaN	NaN
14	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
15	404.0	NaN	NaN	1	1.0	1.0	NaN	NaN
16	NaN	NaN	NaN	1	1.0	NaN	NaN	NaN
17	310.0	312.0	NaN	1	1.0	1.0	1.0	NaN
18	NaN	NaN	NaN	1	NaN	NaN	NaN	NaN
19	NaN	NaN	NaN	1	1.0	NaN	NaN	NaN

2.1.3 Impresora

Las etiquetas generadas por la impresora también se almacenan.

[84]:
```
df_printer = pd.read_csv(r"C:\Users\javier.
 ↪delafuente\Downloads\TFM\openpack\data\datasets\openpack\v0.3.
 ↪1\U0101\system\printer\S0100.csv")
df_printer.head()
```

[84]:

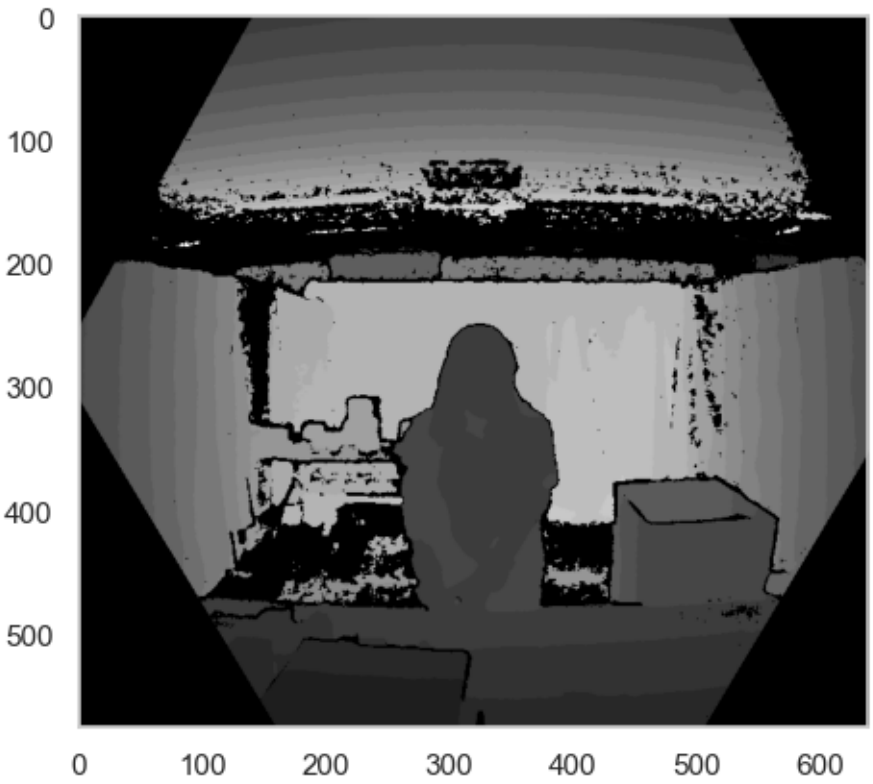
	unixtime	datetime	order_sheet
0	1634178408000	2021-10-14 11:26:48+09:00	DEN010001
1	1634178545000	2021-10-14 11:29:05+09:00	DEN010002
2	1634178628000	2021-10-14 11:30:28+09:00	DEN010003
3	1634178724000	2021-10-14 11:32:04+09:00	DEN010004
4	1634178813000	2021-10-14 11:33:33+09:00	DEN010005

3 Imagen de profundidad

La imagen en profundidad capturada por la cámara Intel® RealSense™ Depth Camera D435i genera en escala de grises una imagen como la que puede verse a continuación, donde se ha amplificado por 10 el valor de cada píxel para facilitar su visualización.

[49]:
```
img=cv2.imread(r"C:\Users\javier.
 ↪delafuente\Downloads\TFM\openpack\depth_data\U0101\kinect\release\frames\depth\S0100\2021\1
 ↪png")
#Displaying image using plt.imshow() method
plt.imshow(img)

#hold the window
plt.imshow(img*10)
plt.grid(False)
```

```
plt.show()
```



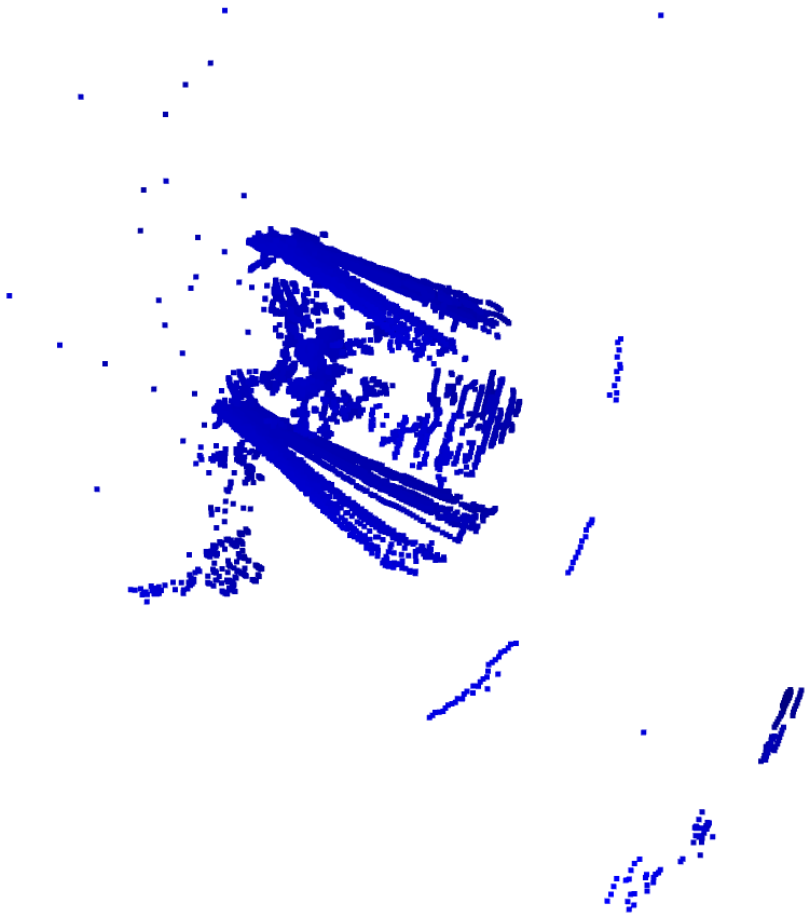
4 LiDAR

Por último, se muestra a continuación una captura de una etiqueta de la nube de puntos extraída con el LiDAR ULTRA PACK(VLP-32C)-Veloidne a partir de la librería OpenPack.

[86]:
```
print("Load a ply point cloud, print it, and render it")
pcd_point_cloud = o3d.data.PCDPointCloud()
pcd = o3d.io.read_point_cloud(r"C:\Users\javier.
  ↪delafuente\Downloads\TFM\openpack\LiDAR_data\2021\11\19\11\20\20211119_112051_408913.
  ↪pcd")
print(pcd)
print(np.asarray(pcd.points))
o3d.visualization.draw_geometries([pcd])
```

```
Load a ply point cloud, print it, and render it
PointCloud with 19392 points.
[[ 0.0038923   0.85773331 -0.22983131]
 [ 0.          0.          0.        ]
 [ 0.00372595  0.76584584 -0.17681153]
```

```
…
[-0.11362857  0.37892395  0.09133013]
[-0.06926653  0.3532176  -0.00628287]
[-0.21912983  2.27679157  0.61288351]]
```



Anexo B

Resultados completos

A continuación se muestran los resultados de inferencia sobre el conjunto de evaluación de cada modelo.

B.1. Resultados IMU

Tabla B.1: Resultados sobre acelerómetro del sensor IMU de muñeca izquierda

name	id	precision	recall	f1	support	key
avg/macro	-1	0,947	0,936	0,941		U0102-S0300
avg/weighted	-1	0,955	0,943	0,949		U0102-S0300
Picking	100	0,952	0,958	0,955	167	U0102-S0300
Relocate Item Label	200	0,971	0,971	0,971	277	U0102-S0300
Assemble Box	300	0,981	0,970	0,975	429	U0102-S0300
Insert Items	400	0,932	0,895	0,913	153	U0102-S0300
Close Box	500	0,948	0,941	0,945	273	U0102-S0300
Attach Box Label	600	0,933	0,916	0,925	107	U0102-S0300
Scan Label	700	0,972	0,934	0,953	335	U0102-S0300
Attach Shipping Label	800	0,913	0,948	0,930	154	U0102-S0300
Put on Back Table	900	0,946	0,898	0,922	118	U0102-S0300
Fill out Order	1000	0,919	0,924	0,921	171	U0102-S0300
avg/macro	-1	0,939	0,936	0,937		U0106-S0300
avg/weighted	-1	0,942	0,940	0,941		U0106-S0300
Picking	100	0,949	0,938	0,943	177	U0106-S0300
Relocate Item Label	200	0,959	0,915	0,936	282	U0106-S0300

Assemble Box	300	0,950	0,963	0,957	356	U0106-S0300
Insert Items	400	0,930	0,920	0,925	174	U0106-S0300
Close Box	500	0,939	0,960	0,949	223	U0106-S0300
Attach Box Label	600	0,926	0,911	0,919	124	U0106-S0300
Scan Label	700	0,953	0,953	0,953	255	U0106-S0300
Attach Shipping Label	800	0,936	0,942	0,939	155	U0106-S0300
Put on Back Table	900	0,942	0,915	0,928	141	U0106-S0300
Fill out Order	1000	0,903	0,946	0,924	167	U0106-S0300
avg/macro	-1	0,894	0,903	0,896		U0210-S0300
avg/weighted	-1	0,925	0,896	0,909		U0210-S0300
Picking	100	0,917	0,951	0,934	245	U0210-S0300
Relocate Item Label	200	0,959	0,873	0,914	322	U0210-S0300
Assemble Box	300	0,875	0,912	0,893	238	U0210-S0300
Insert Items	400	0,930	0,907	0,919	162	U0210-S0300
Close Box	500	0,970	0,815	0,886	200	U0210-S0300
Attach Box Label	600	0,758	0,833	0,794	60	U0210-S0300
Scan Label	700	0,957	0,909	0,933	198	U0210-S0300
Attach Shipping Label	800	0,920	0,913	0,916	138	U0210-S0300
Put on Back Table	900	0,750	1,000	0,857	3	U0210-S0300
Fill out Order	1000	0,908	0,919	0,914	161	U0210-S0300
avg/macro	-1	0,935	0,924	0,930		all
avg/weighted	-1	0,941	0,928	0,935		all
Picking	100	0,936	0,949	0,943	589	all
Relocate Item Label	200	0,963	0,917	0,940	881	all
Assemble Box	300	0,945	0,954	0,949	1023	all
Insert Items	400	0,931	0,908	0,919	489	all
Close Box	500	0,951	0,911	0,930	696	all
Attach Box Label	600	0,891	0,897	0,894	291	all
Scan Label	700	0,962	0,934	0,948	788	all
Attach Shipping Label	800	0,923	0,935	0,929	447	all
Put on Back Table	900	0,941	0,908	0,924	262	all
Fill out Order	1000	0,910	0,930	0,920	499	all

Tabla B.2: Resultados sobre giroscopio del sensor IMU de muñeca izquierda

name	id	precision	recall	f1	support	key
avg/macro	-1	0,938	0,929	0,934		U0102-S0300
avg/weighted	-1	0,948	0,934	0,941		U0102-S0300
Picking	100	0,929	0,946	0,938	167	U0102-S0300
Relocate Item Label	200	0,963	0,939	0,951	277	U0102-S0300
Assemble Box	300	0,974	0,967	0,971	429	U0102-S0300
Insert Items	400	0,897	0,915	0,906	153	U0102-S0300
Close Box	500	0,959	0,938	0,948	273	U0102-S0300
Attach Box Label	600	0,896	0,888	0,892	107	U0102-S0300
Scan Label	700	0,962	0,904	0,932	335	U0102-S0300
Attach Shipping Label	800	0,913	0,955	0,933	154	U0102-S0300
Put on Back Table	900	0,949	0,941	0,945	118	U0102-S0300
Fill out Order	1000	0,939	0,901	0,919	171	U0102-S0300
avg/macro	-1	0,932	0,929	0,930		U0106-S0300
avg/weighted	-1	0,938	0,933	0,935		U0106-S0300
Picking	100	0,918	0,949	0,933	177	U0106-S0300
Relocate Item Label	200	0,967	0,933	0,949	282	U0106-S0300
Assemble Box	300	0,965	0,927	0,946	356	U0106-S0300
Insert Items	400	0,863	0,943	0,901	174	U0106-S0300
Close Box	500	0,959	0,955	0,957	223	U0106-S0300
Attach Box Label	600	0,932	0,879	0,905	124	U0106-S0300
Scan Label	700	0,940	0,976	0,958	255	U0106-S0300
Attach Shipping Label	800	0,942	0,942	0,942	155	U0106-S0300
Put on Back Table	900	0,925	0,872	0,898	141	U0106-S0300
Fill out Order	1000	0,910	0,910	0,910	167	U0106-S0300
avg/macro	-1	0,937	0,896	0,912		U0210-S0300
avg/weighted	-1	0,936	0,920	0,926		U0210-S0300
Picking	100	0,925	0,951	0,938	245	U0210-S0300
Relocate Item Label	200	0,968	0,842	0,900	322	U0210-S0300
Assemble Box	300	0,865	0,916	0,890	238	U0210-S0300
Insert Items	400	0,890	0,951	0,919	162	U0210-S0300
Close Box	500	0,969	0,940	0,954	200	U0210-S0300
Attach Box Label	600	0,864	0,850	0,857	60	U0210-S0300
Scan Label	700	0,974	0,965	0,970	198	U0210-S0300

Attach Shipping Label	800	0,963	0,949	0,956	138	U0210-S0300
Put on Back Table	900	1,000	0,667	0,800	3	U0210-S0300
Fill out Order	1000	0,949	0,925	0,937	161	U0210-S0300
avg/macro	-1	0,935	0,925	0,930		all
avg/weighted	-1	0,940	0,929	0,935		all
Picking	100	0,924	0,949	0,936	589	all
Relocate Item Label	200	0,966	0,901	0,932	881	all
Assemble Box	300	0,944	0,941	0,943	1023	all
Insert Items	400	0,882	0,937	0,909	489	all
Close Box	500	0,962	0,944	0,953	696	all
Attach Box Label	600	0,904	0,876	0,890	291	all
Scan Label	700	0,957	0,943	0,950	788	all
Attach Shipping Label	800	0,938	0,949	0,943	447	all
Put on Back Table	900	0,937	0,901	0,918	262	all
Fill out Order	1000	0,932	0,912	0,922	499	all

Tabla B.3: Resultados sobre acelerómetro del sensor IMU de muñeca derecha

name	id	precision	recall	f1	support	key
avg/macro	-1	0,948	0,940	0,944		U0102-S0300
avg/weighted	-1	0,956	0,946	0,951		U0102-S0300
Picking	100	0,959	0,970	0,964	167	U0102-S0300
Relocate Item Label	200	0,971	0,971	0,971	277	U0102-S0300
Assemble Box	300	0,981	0,963	0,972	429	U0102-S0300
Insert Items	400	0,916	0,928	0,922	153	U0102-S0300
Close Box	500	0,962	0,938	0,950	273	U0102-S0300
Attach Box Label	600	0,922	0,888	0,905	107	U0102-S0300
Scan Label	700	0,966	0,937	0,952	335	U0102-S0300
Attach Shipping Label	800	0,906	0,935	0,920	154	U0102-S0300
Put on Back Table	900	0,940	0,932	0,936	118	U0102-S0300
Fill out Order	1000	0,952	0,936	0,944	171	U0102-S0300
avg/macro	-1	0,931	0,929	0,930		U0106-S0300
avg/weighted	-1	0,936	0,934	0,935		U0106-S0300
Picking	100	0,917	0,932	0,924	177	U0106-S0300

Relocate Item Label	200	0,959	0,904	0,931	282	U0106-S0300
Assemble Box	300	0,953	0,958	0,955	356	U0106-S0300
Insert Items	400	0,906	0,943	0,924	174	U0106-S0300
Close Box	500	0,955	0,951	0,953	223	U0106-S0300
Attach Box Label	600	0,909	0,887	0,898	124	U0106-S0300
Scan Label	700	0,946	0,969	0,957	255	U0106-S0300
Attach Shipping Label	800	0,941	0,923	0,932	155	U0106-S0300
Put on Back Table	900	0,921	0,908	0,914	141	U0106-S0300
Fill out Order	1000	0,905	0,916	0,911	167	U0106-S0300
avg/macro	-1	0,902	0,889	0,894		U0210-S0300
avg/weighted	-1	0,940	0,913	0,926		U0210-S0300
Picking	100	0,933	0,959	0,946	245	U0210-S0300
Relocate Item Label	200	0,962	0,863	0,910	322	U0210-S0300
Assemble Box	300	0,931	0,912	0,921	238	U0210-S0300
Insert Items	400	0,899	0,932	0,915	162	U0210-S0300
Close Box	500	0,978	0,880	0,926	200	U0210-S0300
Attach Box Label	600	0,800	0,867	0,832	60	U0210-S0300
Scan Label	700	0,969	0,955	0,962	198	U0210-S0300
Attach Shipping Label	800	0,935	0,935	0,935	138	U0210-S0300
Put on Back Table	900	0,667	0,667	0,667	3	U0210-S0300
Fill out Order	1000	0,943	0,919	0,931	161	U0210-S0300
avg/macro	-1	0,937	0,928	0,932		all
avg/weighted	-1	0,945	0,932	0,938		all
Picking	100	0,935	0,954	0,945	589	all
Relocate Item Label	200	0,964	0,910	0,936	881	all
Assemble Box	300	0,959	0,949	0,954	1023	all
Insert Items	400	0,907	0,935	0,920	489	all
Close Box	500	0,964	0,925	0,944	696	all
Attach Box Label	600	0,889	0,883	0,886	291	all
Scan Label	700	0,960	0,952	0,956	788	all
Attach Shipping Label	800	0,927	0,931	0,929	447	all
Put on Back Table	900	0,927	0,916	0,921	262	all
Fill out Order	1000	0,933	0,924	0,928	499	all

Tabla B.4: Resultados sobre giroscopio del sensor IMU de muñeca derecha

name	id	precision	recall	f1	support	key
avg/macro	-1	0,930	0,921	0,925		U0102-S0300
avg/weighted	-1	0,941	0,928	0,934		U0102-S0300
Picking	100	0,913	0,940	0,926	167	U0102-S0300
Relocate Item Label	200	0,963	0,931	0,947	277	U0102-S0300
Assemble Box	300	0,961	0,967	0,964	429	U0102-S0300
Insert Items	400	0,884	0,895	0,890	153	U0102-S0300
Close Box	500	0,958	0,927	0,942	273	U0102-S0300
Attach Box Label	600	0,913	0,888	0,900	107	U0102-S0300
Scan Label	700	0,969	0,919	0,943	335	U0102-S0300
Attach Shipping Label	800	0,910	0,922	0,916	154	U0102-S0300
Put on Back Table	900	0,901	0,924	0,912	118	U0102-S0300
Fill out Order	1000	0,927	0,895	0,911	171	U0102-S0300
avg/macro	-1	0,936	0,932	0,934		U0106-S0300
avg/weighted	-1	0,940	0,932	0,936		U0106-S0300
Picking	100	0,939	0,949	0,944	177	U0106-S0300
Relocate Item Label	200	0,963	0,915	0,938	282	U0106-S0300
Assemble Box	300	0,956	0,913	0,934	356	U0106-S0300
Insert Items	400	0,857	0,931	0,893	174	U0106-S0300
Close Box	500	0,943	0,960	0,951	223	U0106-S0300
Attach Box Label	600	0,926	0,903	0,914	124	U0106-S0300
Scan Label	700	0,953	0,953	0,953	255	U0106-S0300
Attach Shipping Label	800	0,948	0,935	0,942	155	U0106-S0300
Put on Back Table	900	0,924	0,943	0,933	141	U0106-S0300
Fill out Order	1000	0,951	0,922	0,936	167	U0106-S0300
avg/macro	-1	0,904	0,929	0,913		U0210-S0300
avg/weighted	-1	0,925	0,913	0,917		U0210-S0300
Picking	100	0,922	0,959	0,940	245	U0210-S0300
Relocate Item Label	200	0,966	0,804	0,878	322	U0210-S0300
Assemble Box	300	0,827	0,903	0,863	238	U0210-S0300
Insert Items	400	0,865	0,951	0,906	162	U0210-S0300
Close Box	500	0,969	0,925	0,946	200	U0210-S0300
Attach Box Label	600	0,873	0,917	0,894	60	U0210-S0300
Scan Label	700	0,980	0,970	0,975	198	U0210-S0300

Attach Shipping Label	800	0,962	0,913	0,937	138	U0210-S0300
Put on Back Table	900	0,750	1,000	0,857	3	U0210-S0300
Fill out Order	1000	0,927	0,944	0,935	161	U0210-S0300
avg/macro	-1	0,930	0,925	0,927		all
avg/weighted	-1	0,936	0,925	0,930		all
Picking	100	0,924	0,951	0,937	589	all
Relocate Item Label	200	0,964	0,880	0,920	881	all
Assemble Box	300	0,925	0,934	0,929	1023	all
Insert Items	400	0,868	0,926	0,896	489	all
Close Box	500	0,956	0,937	0,946	696	all
Attach Box Label	600	0,910	0,900	0,905	291	all
Scan Label	700	0,966	0,943	0,954	788	all
Attach Shipping Label	800	0,939	0,924	0,931	447	all
Put on Back Table	900	0,911	0,935	0,923	262	all
Fill out Order	1000	0,935	0,920	0,927	499	all

B.2. Resultados keypoints

Tabla B.5: Resultados sobre *keypoints* de huesos

name	id	precision	recall	f1	support	key
avg/macro	-1	0,947	0,936	0,941		U0102-S0300
avg/weighted	-1	0,954	0,940	0,947		U0102-S0300
Picking	100	0,915	0,970	0,942	167	U0102-S0300
Relocate Item Label	200	0,992	0,921	0,955	277	U0102-S0300
Assemble Box	300	0,963	0,981	0,972	429	U0102-S0300
Insert Items	400	0,938	0,889	0,913	153	U0102-S0300
Close Box	500	0,956	0,956	0,956	273	U0102-S0300
Attach Box Label	600	0,914	0,897	0,906	107	U0102-S0300
Scan Label	700	0,965	0,907	0,935	335	U0102-S0300
Attach Shipping Label	800	0,925	0,961	0,943	154	U0102-S0300
Put on Back Table	900	0,950	0,958	0,954	118	U0102-S0300
Fill out Order	1000	0,952	0,924	0,938	171	U0102-S0300
avg/macro	-1	0,931	0,927	0,929		U0106-S0300

avg/weighted	-1	0,936	0,926	0,930		U0106-S0300
Picking	100	0,916	0,927	0,921	177	U0106-S0300
Relocate Item Label	200	0,955	0,901	0,927	282	U0106-S0300
Assemble Box	300	0,946	0,933	0,939	356	U0106-S0300
Insert Items	400	0,925	0,925	0,925	174	U0106-S0300
Close Box	500	0,963	0,928	0,945	223	U0106-S0300
Attach Box Label	600	0,891	0,927	0,909	124	U0106-S0300
Scan Label	700	0,933	0,929	0,931	255	U0106-S0300
Attach Shipping Label	800	0,893	0,974	0,932	155	U0106-S0300
Put on Back Table	900	0,969	0,894	0,930	141	U0106-S0300
Fill out Order	1000	0,923	0,928	0,925	167	U0106-S0300
avg/macro	-1	0,839	0,906	0,852		U0210-S0300
avg/weighted	-1	0,906	0,887	0,893		U0210-S0300
Picking	100	0,872	0,918	0,895	245	U0210-S0300
Relocate Item Label	200	0,949	0,755	0,841	322	U0210-S0300
Assemble Box	300	0,872	0,916	0,893	238	U0210-S0300
Insert Items	400	0,927	0,858	0,891	162	U0210-S0300
Close Box	500	0,921	0,875	0,897	200	U0210-S0300
Attach Box Label	600	0,815	0,883	0,848	60	U0210-S0300
Scan Label	700	0,956	0,980	0,968	198	U0210-S0300
Attach Shipping Label	800	0,915	0,935	0,925	138	U0210-S0300
Put on Back Table	900	0,300	1,000	0,462	3	U0210-S0300
Fill out Order	1000	0,859	0,944	0,899	161	U0210-S0300
avg/macro	-1	0,927	0,921	0,923		all
avg/weighted	-1	0,933	0,920	0,926		all
Picking	100	0,897	0,935	0,916	589	all
Relocate Item Label	200	0,965	0,854	0,906	881	all
Assemble Box	300	0,935	0,949	0,942	1023	all
Insert Items	400	0,930	0,892	0,910	489	all
Close Box	500	0,948	0,924	0,936	696	all
Attach Box Label	600	0,883	0,907	0,895	291	all
Scan Label	700	0,952	0,933	0,942	788	all
Attach Shipping Label	800	0,911	0,957	0,933	447	all
Put on Back Table	900	0,934	0,924	0,929	262	all
Fill out Order	1000	0,910	0,932	0,921	499	all

Tabla B.6: Resultados sobre *keypoints* de articulaciones

name	id	precision	recall	f1	support	key
avg/macro	-1	0,940	0,937	0,938		U0102-S0300
avg/weighted	-1	0,950	0,941	0,945		U0102-S0300
Picking	100	0,907	0,994	0,949	167	U0102-S0300
Relocate Item Label	200	0,989	0,942	0,965	277	U0102-S0300
Assemble Box	300	0,970	0,967	0,968	429	U0102-S0300
Insert Items	400	0,893	0,928	0,910	153	U0102-S0300
Close Box	500	0,948	0,941	0,945	273	U0102-S0300
Attach Box Label	600	0,904	0,879	0,891	107	U0102-S0300
Scan Label	700	0,972	0,916	0,943	335	U0102-S0300
Attach Shipping Label	800	0,920	0,968	0,943	154	U0102-S0300
Put on Back Table	900	0,956	0,915	0,935	118	U0102-S0300
Fill out Order	1000	0,940	0,918	0,929	171	U0102-S0300
avg/macro	-1	0,937	0,935	0,936		U0106-S0300
avg/weighted	-1	0,942	0,937	0,939		U0106-S0300
Picking	100	0,949	0,944	0,946	177	U0106-S0300
Relocate Item Label	200	0,923	0,940	0,931	282	U0106-S0300
Assemble Box	300	0,968	0,933	0,950	356	U0106-S0300
Insert Items	400	0,921	0,937	0,929	174	U0106-S0300
Close Box	500	0,955	0,951	0,953	223	U0106-S0300
Attach Box Label	600	0,890	0,911	0,900	124	U0106-S0300
Scan Label	700	0,960	0,945	0,953	255	U0106-S0300
Attach Shipping Label	800	0,954	0,929	0,941	155	U0106-S0300
Put on Back Table	900	0,921	0,915	0,918	141	U0106-S0300
Fill out Order	1000	0,929	0,946	0,938	167	U0106-S0300
avg/macro	-1	0,929	0,887	0,904		U0210-S0300
avg/weighted	-1	0,922	0,918	0,919		U0210-S0300
Picking	100	0,947	0,947	0,947	245	U0210-S0300
Relocate Item Label	200	0,942	0,916	0,929	322	U0210-S0300
Assemble Box	300	0,796	0,933	0,859	238	U0210-S0300
Insert Items	400	0,920	0,778	0,843	162	U0210-S0300
Close Box	500	0,955	0,950	0,952	200	U0210-S0300
Attach Box Label	600	0,898	0,883	0,891	60	U0210-S0300
Scan Label	700	0,974	0,960	0,967	198	U0210-S0300
Attach Shipping Label	800	0,940	0,906	0,923	138	U0210-S0300

Put on Back Table	900	1,000	0,667	0,800	3	U0210-S0300
Fill out Order	1000	0,920	0,932	0,926	161	U0210-S0300
avg/macro	-1	0,934	0,928	0,931		all
avg/weighted	-1	0,938	0,933	0,935		all
Picking	100	0,935	0,959	0,947	589	all
Relocate Item Label	200	0,950	0,932	0,941	881	all
Assemble Box	300	0,923	0,947	0,935	1023	all
Insert Items	400	0,911	0,881	0,896	489	all
Close Box	500	0,952	0,947	0,950	696	all
Attach Box Label	600	0,897	0,893	0,895	291	all
Scan Label	700	0,969	0,937	0,952	788	all
Attach Shipping Label	800	0,937	0,935	0,936	447	all
Put on Back Table	900	0,937	0,912	0,925	262	all
Fill out Order	1000	0,930	0,932	0,931	499	all

Tabla B.7: Resultados sobre *keypoints* de huesos en movimiento

name	id	precision	recall	f1	support	key
avg/macro	-1	0,880	0,882	0,880		U0102-S0300
avg/weighted	-1	0,892	0,885	0,888		U0102-S0300
Picking	100	0,850	0,952	0,898	167	U0102-S0300
Relocate Item Label	200	0,905	0,895	0,900	277	U0102-S0300
Assemble Box	300	0,915	0,900	0,907	429	U0102-S0300
Insert Items	400	0,835	0,863	0,849	153	U0102-S0300
Close Box	500	0,853	0,850	0,851	273	U0102-S0300
Attach Box Label	600	0,821	0,860	0,840	107	U0102-S0300
Scan Label	700	0,947	0,910	0,928	335	U0102-S0300
Attach Shipping Label	800	0,898	0,857	0,877	154	U0102-S0300
Put on Back Table	900	0,845	0,924	0,883	118	U0102-S0300
Fill out Order	1000	0,926	0,807	0,862	171	U0102-S0300
avg/macro	-1	0,867	0,862	0,864		U0106-S0300
avg/weighted	-1	0,873	0,867	0,869		U0106-S0300
Picking	100	0,835	0,887	0,860	177	U0106-S0300
Relocate Item Label	200	0,879	0,901	0,890	282	U0106-S0300

Assemble Box	300	0,896	0,874	0,885	356	U0106-S0300
Insert Items	400	0,807	0,868	0,837	174	U0106-S0300
Close Box	500	0,889	0,865	0,877	223	U0106-S0300
Attach Box Label	600	0,828	0,895	0,860	124	U0106-S0300
Scan Label	700	0,900	0,886	0,893	255	U0106-S0300
Attach Shipping Label	800	0,863	0,852	0,857	155	U0106-S0300
Put on Back Table	900	0,886	0,773	0,826	141	U0106-S0300
Fill out Order	1000	0,884	0,820	0,851	167	U0106-S0300
avg/macro	-1	0,739	0,823	0,738		U0210-S0300
avg/weighted	-1	0,810	0,785	0,792		U0210-S0300
Picking	100	0,749	0,718	0,733	245	U0210-S0300
Relocate Item Label	200	0,874	0,649	0,745	322	U0210-S0300
Assemble Box	300	0,741	0,756	0,748	238	U0210-S0300
Insert Items	400	0,673	0,877	0,761	162	U0210-S0300
Close Box	500	0,806	0,810	0,808	200	U0210-S0300
Attach Box Label	600	0,845	0,817	0,831	60	U0210-S0300
Scan Label	700	0,885	0,929	0,906	198	U0210-S0300
Attach Shipping Label	800	0,883	0,819	0,850	138	U0210-S0300
Put on Back Table	900	0,070	1,000	0,130	3	U0210-S0300
Fill out Order	1000	0,868	0,857	0,863	161	U0210-S0300
avg/macro	-1	0,844	0,850	0,846		all
avg/weighted	-1	0,856	0,850	0,852		all
Picking	100	0,807	0,835	0,821	589	all
Relocate Item Label	200	0,887	0,807	0,845	881	all
Assemble Box	300	0,867	0,857	0,862	1023	all
Insert Items	400	0,764	0,869	0,813	489	all
Close Box	500	0,851	0,843	0,847	696	all
Attach Box Label	600	0,829	0,866	0,847	291	all
Scan Label	700	0,915	0,907	0,911	788	all
Attach Shipping Label	800	0,881	0,843	0,862	447	all
Put on Back Table	900	0,749	0,844	0,794	262	all
Fill out Order	1000	0,892	0,828	0,859	499	all

Tabla B.8: Resultados sobre *keypoints* de articulaciones en movimiento

name	id	precision	recall	f1	support	key
avg/macro	-1	0,900	0,903	0,901		U0102-S0300
avg/weighted	-1	0,908	0,903	0,905		U0102-S0300
Picking	100	0,896	0,928	0,912	167	U0102-S0300
Relocate Item Label	200	0,929	0,892	0,910	277	U0102-S0300
Assemble Box	300	0,953	0,902	0,927	429	U0102-S0300
Insert Items	400	0,861	0,889	0,875	153	U0102-S0300
Close Box	500	0,833	0,930	0,879	273	U0102-S0300
Attach Box Label	600	0,875	0,850	0,863	107	U0102-S0300
Scan Label	700	0,934	0,887	0,910	335	U0102-S0300
Attach Shipping Label	800	0,858	0,903	0,880	154	U0102-S0300
Put on Back Table	900	0,917	0,941	0,929	118	U0102-S0300
Fill out Order	1000	0,945	0,912	0,929	171	U0102-S0300
avg/macro	-1	0,876	0,876	0,875		U0106-S0300
avg/weighted	-1	0,882	0,879	0,879		U0106-S0300
Picking	100	0,861	0,876	0,868	177	U0106-S0300
Relocate Item Label	200	0,898	0,901	0,899	282	U0106-S0300
Assemble Box	300	0,887	0,902	0,894	356	U0106-S0300
Insert Items	400	0,860	0,885	0,873	174	U0106-S0300
Close Box	500	0,859	0,821	0,839	223	U0106-S0300
Attach Box Label	600	0,814	0,919	0,864	124	U0106-S0300
Scan Label	700	0,950	0,894	0,921	255	U0106-S0300
Attach Shipping Label	800	0,857	0,929	0,892	155	U0106-S0300
Put on Back Table	900	0,894	0,780	0,833	141	U0106-S0300
Fill out Order	1000	0,877	0,850	0,863	167	U0106-S0300
avg/macro	-1	0,761	0,856	0,772		U0210-S0300
avg/weighted	-1	0,850	0,827	0,833		U0210-S0300
Picking	100	0,898	0,755	0,820	245	U0210-S0300
Relocate Item Label	200	0,917	0,720	0,807	322	U0210-S0300
Assemble Box	300	0,818	0,849	0,833	238	U0210-S0300
Insert Items	400	0,799	0,858	0,827	162	U0210-S0300
Close Box	500	0,889	0,840	0,864	200	U0210-S0300
Attach Box Label	600	0,725	0,833	0,775	60	U0210-S0300
Scan Label	700	0,891	0,909	0,900	198	U0210-S0300

Attach Shipping Label	800	0,769	0,870	0,816	138	U0210-S0300
Put on Back Table	900	0,136	1,000	0,240	3	U0210-S0300
Fill out Order	1000	0,768	0,925	0,839	161	U0210-S0300
avg/macro	-1	0,866	0,873	0,869		all
avg/weighted	-1	0,878	0,873	0,875		all
Picking	100	0,886	0,840	0,862	589	all
Relocate Item Label	200	0,914	0,832	0,871	881	all
Assemble Box	300	0,897	0,890	0,893	1023	all
Insert Items	400	0,840	0,877	0,858	489	all
Close Box	500	0,856	0,869	0,862	696	all
Attach Box Label	600	0,815	0,876	0,844	291	all
Scan Label	700	0,928	0,895	0,911	788	all
Attach Shipping Label	800	0,829	0,902	0,864	447	all
Put on Back Table	900	0,842	0,855	0,848	262	all
Fill out Order	1000	0,858	0,896	0,876	499	all

B.3. Resultados finales

Tabla B.9: Resultados sobre el modelo unificado

name	id	precision	recall	f1	support	key
avg/macro	-1	0,972	0,951	0,962		U0102-S0300
avg/weighted	-1	0,939	0,925	0,932		U0102-S0300
Picking	100	0,998	0,929	0,962	167	U0102-S0300
Relocate Item Label	200	0,967	0,991	0,979	277	U0102-S0300
Assemble Box	300	0,954	0,903	0,928	429	U0102-S0300
Insert Items	400	0,967	0,891	0,927	153	U0102-S0300
Close Box	500	0,912	0,932	0,922	273	U0102-S0300
Attach Box Label	600	0,877	0,850	0,863	107	U0102-S0300
Scan Label	700	0,996	0,913	0,953	335	U0102-S0300
Attach Shipping Label	800	0,899	0,926	0,913	154	U0102-S0300
Put on Back Table	900	0,973	0,943	0,958	118	U0102-S0300
Fill out Order	1000	0,985	0,914	0,948	171	U0102-S0300
avg/macro	-1	0,877	0,878	0,878		U0106-S0300
avg/weighted	-1	0,921	0,881	0,900		U0106-S0300

Picking	100	0,949	0,878	0,912	177	U0106-S0300
Relocate Item Label	200	0,899	0,989	0,942	282	U0106-S0300
Assemble Box	300	0,889	0,954	0,920	356	U0106-S0300
Insert Items	400	0,861	0,887	0,874	174	U0106-S0300
Close Box	500	0,924	0,927	0,926	223	U0106-S0300
Attach Box Label	600	0,830	0,920	0,873	124	U0106-S0300
Scan Label	700	0,952	0,895	0,923	255	U0106-S0300
Attach Shipping Label	800	0,860	0,931	0,894	155	U0106-S0300
Put on Back Table	900	0,976	0,933	0,954	141	U0106-S0300
Fill out Order	1000	0,930	0,918	0,924	167	U0106-S0300
avg/macro	-1	0,837	0,978	0,902		U0210-S0300
avg/weighted	-1	0,990	0,912	0,949		U0210-S0300
Picking	100	0,950	0,812	0,875	245	U0210-S0300
Relocate Item Label	200	0,919	0,808	0,860	322	U0210-S0300
Assemble Box	300	0,821	0,998	0,901	238	U0210-S0300
Insert Items	400	0,856	0,999	0,922	162	U0210-S0300
Close Box	500	0,932	0,842	0,885	200	U0210-S0300
Attach Box Label	600	0,768	0,911	0,833	60	U0210-S0300
Scan Label	700	0,894	0,911	0,902	198	U0210-S0300
Attach Shipping Label	800	0,811	0,924	0,864	138	U0210-S0300
Put on Back Table	900	0,185	1,000	0,313	3	U0210-S0300
Fill out Order	1000	0,876	0,926	0,900	161	U0210-S0300
avg/macro	-1	0,962	0,961	0,961		all
avg/weighted	-1	0,963	0,96	0,961		all
Picking	100	0,97	0,953	0,961	589	all
Relocate Item Label	200	0,965	0,954	0,959	881	all
Assemble Box	300	0,915	0,929	0,922	1023	all
Insert Items	400	0,99	0,921	0,954	489	all
Close Box	500	0,928	0,899	0,913	696	all
Attach Box Label	600	0,903	0,907	0,905	291	all
Scan Label	700	0,945	0,993	0,968	788	all
Attach Shipping Label	800	0,941	0,959	0,950	447	all
Put on Back Table	900	0,939	0,934	0,936	262	all
Fill out Order	1000	0,943	0,936	0,939	499	all