SEGUIMIENTO Y EVALUACIÓN DE PERSONAS EN AMBIENTES CERRADOS / ABIERTOS

Bryan Santiago Cruz Angel

Trabajo Dirigido

Tutor
Oscar Julián Perdomo Charry
PhD en Ingeniería - Sistemas y Computación
M.Sc en Ingeniería Eléctrica
Ing. Electrónico

UNIVERSIDAD DEL ROSARIO
ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO
PROGRAMA DE INGENIERÍA BIOMÉDICA
BOGOTÁ D.C
2021

RESUMEN

Actualmente, en un mundo constantemente en evolución hemos evidenciado el impacto y crecimiento de la tecnología. Lo que se conoce como la revolución tecnológica ha llevado a la creación de sistemas de pago en línea, Inteligencia artificial (IA), BigData, el internet de las cosas, entre otros. Estas herramientas facilitan y optimizan procesos que normalmente generan más gastos económicos, ambientales y sociales, en especial el uso de tecnologías de IA presenta un gran desempeño en la detección de objetos por medio de imágenes, cuando se trata de seguimiento de personas, la norma general dicta que debe realizarse de forma manual, en donde un trabajador sea el encargado de llevar un control de las cámaras de vigilancia, mediante las cuales se puede identificar un sujeto, su posición, y su estado físico, este control es efectivo pero tedioso, en el enfoque de este trabajo dirigido se busca automatizar dicha labor especialmente en centros de salud o ancianatos, donde pacientes con condiciones neurológicas necesitan atención o monitoreo más constante, apoyado en el algoritmo de Detección de objetos YOLO en su versión 3 y en el algoritmo DeepSORT para el seguimiento de personas, se plantea generar un programa en Python de detección y evaluación de actividad física de personas en tiempo real con base en diversas técnicas de IA.

Estos algoritmos tienen diversos usos y ventajas, algunos ejemplos van desde el seguimiento del balón en partidos de futbol o baloncesto hasta la detección de automóviles para sistemas de conducción automática, entre las ventajas que ofrece es que es una tecnología no invasiva, de bajo costo y alta eficiencia, por otra parte, presenta algunas dificultades relacionadas con efectos como la oclusión, que es cuando una persona se encuentra delante de otra y dificulta su visibilidad en el video. La gran cantidad de personas puede ser otro limitante, ya que por lo general se requiere de gran capacidad computacional para detectar cada objeto, esto de forma intrínseca nos afecta el rendimiento en tiempo real, limitando el número de fotogramas que se pueden procesar cada segundo.

AGRADECIMIENTOS

Primero un agradecimiento a Dios, quien fue el que me dio a mi familia, mis amigos, compañeros y demás personas que finalmente colocaron su granito de esperanza en mí, a todos los que estuvieron desde que era un niño y siguen estando presentes hoy en mi corazón, a ustedes les dedico este trabajo dirigido, especialmente agradezco a mi madre, mi padre, mis abuelitas, mis hermanas, tías, tíos y demás familiares les agradezco por estar siempre a mi lado y acompañarme en esta importante etapa de mi vida, sin ustedes no habría podido concluir con este proceso de formación, sin su ayuda incondicional, son mi motor de vida y les debo todo lo que soy.

Un gran agradecimiento a mi tutor, el ingeniero Oscar Julián Perdomo Charry por permitirme ser parte de este proyecto, por confiar en mí y estar siempre presente ante cualquier dificultad, por acompañarme en el desarrollo y finalización de este trabajo dirigido, por su esfuerzo, este trabajo le pertenece a usted también.

Agradecer a los autores, estudiantes, profesionales o simplemente apasionados como yo, a que el conocimiento sea universal y esté disponible para todo el mundo, agradezco a toda la comunidad científica, pero destaco a los autores de los canales de YouTube *Python Lessons y AMP Tech*, ya que sin sus tutoriales, repositorios e información no hubiera sido posible desarrollar un algoritmo con las características necesarias para el proyecto.

Quiero agradecer finalmente a las dos instituciones que durante los 5 años más importantes y significativos de mi vida me han otorgado los conocimientos, experiencias y carácter que hoy en día me caracterizan como ser humano. A ustedes Escuela Colombiana de Ingeniería Julio Garavito y Universidad del Rosario, les debo todo el orgullo de ser un estudiante de Ingeniería Biomédica, en un futuro espero contribuir a la sociedad con todo lo valioso que me han enseñado para mi vida profesional y personal

Índice General

Índice de Tablas

Índice de figuras

Capítulo 1 - Introducción

Para dar inicio con el trabajo dirigido, es de suma importancia entender el origen de la problemática, así como los temas relacionados a la misma, los cuales se encuentran en este capítulo, en un principio la idea de generar un algoritmo para la detección y seguimiento de personas surge de las dificultades que representa tener personal humano encargado de observar largos videos de cámaras de seguridad, debido a sus errores y el tiempo que toma realizar este tipo de tareas. Lo anterior en un ámbito clínico, donde personas con alteraciones en funciones cognitivas deben ser monitoreadas constantemente. Se ha visto el gran impacto que está generando la inteligencia artificial hoy en día, en especial tecnologías de visión por computadora muestran buen rendimiento en aplicaciones de detección y seguimiento de personas en tiempo real, por ello se realiza la definición de los objetivos en el capítulo 2 de este trabajo. Además, en el capítulo 3 se lleva a cabo la investigación y se definen algunos conceptos o métodos para facilitar el entendimiento de la solución a la cual se desea llegar.

Mas adelante en el documento, hablaremos también sobre los métodos seleccionados y las bases de datos que se trabajaron, de esta forma finalmente se llegara a establecer un algoritmo para detección y seguimiento que se define en el capítulo 4 de metodología, el cual nos arroja resultados en el capítulo 5 y 6, estos resultados demuestran cierto rendimiento y herramientas graficas que permiten a profesionales fisioterapeutas identificar trayectorias, velocidad y gasto calórico de los pacientes en el video, finalmente se darán las conclusiones y posibles mejoras para el futuro del proyecto en el capítulo 7 y 8 respectivamente.

Para comenzar revisaremos el impacto que tiene hoy en día la inteligencia artificial, en diferentes sectores, de forma que se entienda de mejor forma sus ventajas en una posible aplicación médica.

En la actualidad, en un mundo donde los seres humanos controlan todos los procesos y se encuentran gobernando sobre cada parte del planeta, uno de los mayores miedos que a su vez están generando diversos impactos sociales es la evolución de la Inteligencia artificial, llevándonos a planteamientos nunca vistos en esta época, considerada la 4ta revolución industrial. En este contexto, algunas preguntas que surgen son ¿Los robots con IA dominarán el mundo?, ¿Que sucede en los sectores laborales?, ¿Puede llegar a tener conciencia una IA? Entre muchas otras. A continuación, discutiremos algunos de los efectos sociales y respuestas a estas preguntas, viéndolo desde mi punto de vista y comparándolo con algunos autores.

Lo primero, la inteligencia artificial depende meramente del humano, de los accesos que se le otorgan, las tareas asignadas y las herramientas para elaborar dichas tareas, por ende no es común que la IA se salga de sus capacidades para realizar otras tareas, debemos comparar qué nos diferencia a nosotros de un sistema computacional hipercomplejo, ya que en realidad el futuro es incierto, pues así como todavía no entendemos ni siquiera qué es la inteligencia en el ser humano (existe inteligencia musical, espacial, emocional, etc.), no podemos llegar a entender el método completo de procesamiento dentro de la IA, pues de cierta forma le decimos cómo aprender, pero ella interpreta los datos para cumplir de la mejor forma su labor según modelos muy complejos y abstractos, por ende no sabemos

qué decisión va a tomar en cada ejecución, de allí que se considere IA y no solo una serie de instrucciones en un programa. [1]

Respecto al impacto social, ya este se observa en el día a día, en las decisiones que tomamos en base a programas de recomendación basados en IA, en las redes sociales y en varios otros campos que nos afectan directa o indirectamente, como en áreas de diagnóstico médico o el armamento militar, son varios los efectos sociales positivos y negativos que genera la inteligencia artificial. [2]

Para el impacto económico y laboral, la IA puede representar la pérdida de cientos de miles de trabajos, no solo simples como ensamblar una botella, sino tan complejos como el arte, la medicina, conducción de autos y muchos más, aunque esto debemos verlo como una transformación, un cambio a nuevos empleos, no solo la destrucción de un estilo de vida, sino la creación de otros varios, con cada día más comodidades gracias a la tecnología. [3]

Esta tecnología basada en IA resulta revolucionaria e inevitable, con la digitalización y automatización de todos los sistemas del mundo moderno, se ha visto una gran oportunidad para utilizar herramientas informáticas que permitan agilizar procesos como es el caso del reconocimiento de personas por medio de videos, utilizar la inteligencia artificial ofrece varias ventajas, como una alta eficiencia y un costo de operación e implementación bastante bajo. [4] [5]

El objetivo principal de este trabajo dirigido es hacer un algoritmo basado en inteligencia artificial (IA), que use los videos de vigilancia vistos desde una computadora para el monitoreo y cuantificación del desplazamiento de personas y cantidad calórica, este algoritmo debe servir como herramienta para la toma de decisiones y generación de alertas enfocado en el cuidado de las personas y la ayuda al personal médico, así como también evita la utilización de dispositivos sobre el paciente y permite el monitoreo de un gran número de personas en tiempo real.

1.1. Problemática

Actualmente, entre las fuentes de datos que aportan terabytes de Big data tenemos los datos de videovigilancia, debido a la gran cantidad de cámaras de vigilancia en zonas residenciales, instituciones educativas, bancos, hospitales y empresas. Para el análisis de estos videos se emplea personal humano y requiere el análisis de largos videos para identificar algún patrón, por lo que analizarlos no es una tarea fácil, ya que implica el reconocimiento de objetos, el reconocimiento de eventos y su clasificación como normales o anormales. Con esta información se puede generar una alerta en casos de emergencia, así como dar la ubicación actual y monitorear la actividad física que realiza el paciente en los centros de salud. [6] Sobre esto nos enfocamos especialmente en el contexto clínico que se profundiza en el siguiente ítem (1.2), el cual fundamenta la necesidad en centros de salud, psiquiatría o rehabilitación, basándonos en la pregunta de investigación ¿Cómo se realiza el seguimiento y detección de personas por medio del análisis de videos?.

Aunque cabe resaltar que la actividad física principalmente se refiere a caminar, ya que actividades diferentes como saltar, agacharse, levantamiento de pesas y demás, no son cuantificables a partir de la posición del paciente, para dichas actividades se requiere un análisis diferente de eventos a partir de la posición de las extremidades del paciente y no se su posición geográfica en el espacio 3D.

1.2. Contexto clínico

Hoy en día, con el uso la tecnología biomédica en centros de salud se puede llegar a un mejor diagnóstico y monitoreo de los pacientes, especialmente cuando hablamos de monitoreo de pacientes con problemas neurológicos, pacientes de edad avanzada o pacientes en postoperatorio. De hecho, algunos de estos pacientes requieren cumplir con rutinas diarias de ejercicio físico, y conocer la posición actual del paciente puede resultar muy útil para la generación de respuestas en tiempo real en caso de emergencia [7]. Por ejemplo, los pacientes que sufren de Alzheimer, ya que, debido a su condición neurodegenerativa y alteraciones en funciones cognitivas como memoria a corto plazo, nivel de conciencia y orientación, pueden alejarse del centro de salud y perderse de la vista del personal médico, lo cual puede conducir al peligro. Por lo tanto, los pacientes con Alzheimer en ambientes abiertos deben ser monitoreados con mayor atención para garantizar su seguridad. [8]

El monitoreo realizado por fisioterapeutas se enfoca en el seguimiento de la posición y movimientos de los pacientes en terapia física, ya que algunas personas con discapacidad física requieren hacer ejercicios repetitivos y poco motivadores . Existen personas que no tienen una supervisión cuando realizan los ejercicios en casa y esto puede perjudicar su salud en vez de mejorarla. Tenemos la limitante que el personal de rehabilitación no puede monitorear a más de una persona en tiempo real y esta tarea es aún más difícil o tediosa cuando se incrementa la cantidad de personas o el tiempo de captura del video [9]. Por ello se ha visto la necesidad de crear alternativas para el seguimiento de personas, como los rastreadores GPS en una pulsera, la utilización de un móvil con *bluetooth*, la instalación de sensores y marcadores para identificar al paciente. Estas tecnologías poseen problemas relacionadas con el consumo de batería asociado y que los pacientes deben tenerlas colocadas todo el tiempo para garantizar un correcto funcionamiento, esto resulta incómodo para el paciente y ocasiona la pérdida del dispositivo. [8] De aquí que una de las mejores alternativas no invasivas sea el uso de IA por medio de videovigilancia.

1.3. Que es la Inteligencia Artificial (IA)

La inteligencia artificial es la combinación de algoritmos que permitan a las máquinas pensar como el ser humano, que de cierta forma tengan la capacidad de aprender, de razonar y resolver problemas en base a su conocimiento. Estos programas informáticos basados en IA son diseñados para desempeñarse en un tema específico, pueden autocorregirse para mejorar su desempeño en una tarea asignada, lo ideal es que puedan entender y reaccionar con su entorno. La principal diferencia entre la IA y un programa informático común para la toma de decisiones, es principalmente su capacidad de aprender, pues en el caso de un programa simple para tomar decisiones se sigue una serie de pasos específicos para ejecutar su tarea, en cambio, el programa con IA basado en aprendizaje automático, es capaz de llegar entender cómo solucionar su tarea de forma autónoma, no le decimos como hacer su tarea, simplemente se le da información para que él mismo aprenda si con su forma de “pensar” está llegando a la solución correcta. [10]

La IA allana el camino para que las computadoras piensen como humanos, de forma que puedan aplicar conocimientos o experiencias previas para la automatización de tareas y la

toma de decisiones. La IA ha logrado ser aplicada en diferentes campos o sectores como: salud, agricultura, industria química, la industria del petróleo y carbón, transporte, construcción, defensa, vigilancia y seguridad. [4]

En el sector salud, la IA se usa como herramienta de apoyo para el diagnóstico, tratamiento, creación de fármacos, medicina personalizada, monitorización y cuidado del paciente [5]. Lo anterior, gracias a la velocidad de procesamiento y precisión de sus tareas, ayudando en la minimización del error humano al apoyarse en técnicas que permiten a la computadora adquirir experiencia con el objetivo de mejorar su desempeño, como es el *Machine Learning* o aprendizaje automático. En nuestro enfoque resulta sumamente útil en una de las ramas de la IA, esta es la visión por computador, desde aquí con algoritmos de IA se logra obtener información sobre videos e imágenes, que puede ayudar al diagnóstico y monitorización de pacientes. [11]

1.4. ¿Como se crea una inteligencia artificial?

Una de las características de la inteligencia artificial es que tiene diferentes formas de aprendizaje, pero lo que tienen en común es quien les enseña, el ser humano, por ende, pueden presentar los mismos sesgos o errores que presente el conjunto de datos de entrenamiento. Un claro ejemplo se presentó en el 2018, cuando *Amazon* tuvo que descartar una inteligencia artificial de contratación entrenada con solicitudes de empleo que la empresa había recibido a lo largo de 10 años porque el programa, fiel a la historia del sector tecnológico, discriminaba sin justificación a las mujeres. [12]

Respecto a la elaboración de una inteligencia artificial tenemos que lograr que el sistema computacional emule como piensa un humano. De aquí que una maquina inteligente ideal sea flexible, y que a medida que transcurre el tiempo y aumente el número de ejemplos que la IA conozca, mejore sus probabilidades de éxito para predecir o clasificar dependiendo para lo que sea entrenada.

1.5. Redes Neuronales Artificiales

Son modelos computacionales que se inspiran en el comportamiento del cerebro biológicamente hablando. Entiéndase por redes computacionales aquellas en las que existen elementos procesadores de información en donde sus interacciones locales generan a su vez un comportamiento conjunto de todo el sistema, así como el cerebro humano su aprendizaje está basado en experiencias y extracción de características o información.

Las redes neuronales artificiales (RNA) poseen un elemento básico resultado de la analogía con el cerebro, este elemento es conocido como neurona artificial, cada neurona o procesador elemental recibe y pondera la información a sus entradas para producir un valor a su salida, para el aprendizaje de la red se va modificando dicha ponderación en unas series de iteraciones o entrenamientos consecutivos hasta alcanzar el desempeño requerido.
La Figura 1, presenta el concepto de neurona, conjunto de neuronas o capas, las diferentes conexiones para conformar una red y como es empleada como un modelo. [13] [14]

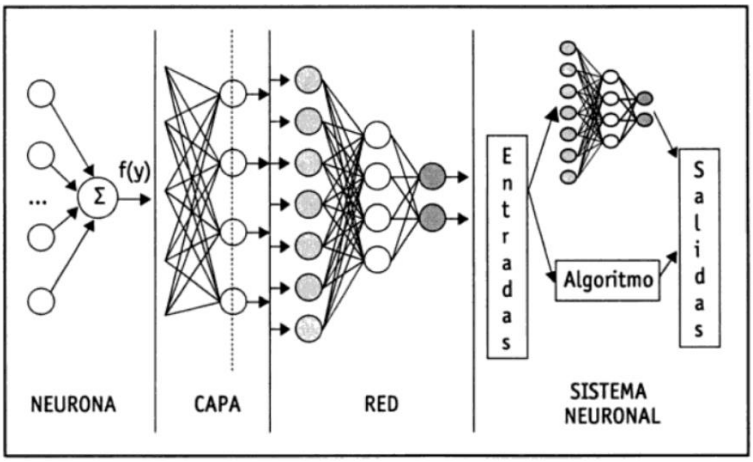

Figura 1. Estructura jerárquica de un sistema neuronal. [14]

1.5.1. Aprendizaje automático

El aprendizaje automático conocido en inglés como *Machine Learning* es una rama de la inteligencia artificial que busca crear algoritmos que tengan la capacidad de aprender e inferir de los datos, sin la necesidad de programar manualmente todos los escenarios posibles. El aprendizaje automático busca obtener la mayor cantidad de información e identifica patrones, similitudes o rasgos que permitan un aprendizaje de los datos, lo que se conoce como entrenamiento del modelo. Luego del entrenamiento, se realiza una etapa de prueba, en donde lo que se busca es darle entradas no conocidas al algoritmo para que nos arroje predicciones: si las predicciones son correctas el algoritmo aprendió de forma correcta, si no obtuvo buenos resultados probablemente necesita más data o un cambio en el modo de entrenamiento. Los procesos de entrenamiento y prueba son realizados de manera iterativa y exploratoria buscando optimizar el desempeño del algoritmo en pro de aumentar la precisión en la toma de decisiones. [15] [16]

Posterior al entrenamiento, el algoritmo ya identificó las características con mayor relevancia para la toma de decisiones, es aquí cuando el algoritmo debe guardar estas reglas o pesos para cada variable dentro de un modelo definido, es decir, que ya se ha generado un pensamiento estructurado para la toma de decisiones y este es generalmente el que termina utilizando el usuario.

1.5.1.1. Aprendizaje supervisado

El aprendizaje supervisado es análogo a estudiar conociendo las preguntas y las respuestas correctas, para que en base a ello pueda generar respuesta futuras o predicciones, es decir, se le están dando ejemplos para que encuentre la relación entre las características de entrada y los resultados deseados. Como ejemplos tenemos los algoritmos basados en regresión lineal, redes neuronales, regresión polinómica, arboles de decisión, entre otros. [3]
Un ejemplo que ayuda a entender mejor el proceso de aprendizaje supervisado es el entrenamiento de un robot para lanzar flechas y acertar al blanco. Los datos serían las

flechas, el modelo sería el mecanismo para lanzar flechas, la función objetiva seria calcular que tan lejos se disparan las flechas del objetivo. Si la máquina se equivoca y no acierta el blanco, un algoritmo de optimización interviene para ajustar el proceso computacional y la máquina o computadora repite el proceso una gran cantidad de veces hasta obtener una respuesta correcta y mejorar. [3]

1.5.2. Aprendizaje profundo

El aprendizaje profundo o en inglés *Deep Learning*, se puede considerar como la evolución o el siguiente nivel del *Machine Learning*. El aprendizaje profundo es uno de los métodos de inteligencia artificial más usados en la actualidad, con la particularidad de poseer una de varias capas para la extracción de características para representación y caracterización de los datos, seguida de un conjunto de capas para la clasificación automática de las características previamente extraídas. Algunos ejemplos de algoritmos basados en aprendizaje profundo son las redes neuronales convolucionales, redes neuronales profundas y redes de creencia profunda, las cuales han demostrado buen desempeño para tareas como el reconocimiento de voz, procesamiento de lenguaje natural, visión del computador, coches que se manejan de manera autónoma, entre otros. [17] [3]

1.6. Google Colaboratory

Es un entorno de programación basado en los notebooks de *Jupyter*, su principal característica es que utiliza máquinas virtuales, es decir que no utiliza nuestros recursos computacionales, sino que nos facilita de forma gratuita un procesamiento en la nube de GPU o TPU dispuesto en un servidor online.
Una de sus grandes ventajas es que podemos editar nuestro código inclusive desde el móvil, ya que guarda nuestro progreso directamente en la Nube de Google y se ejecuta desde el explorador, sin necesidad de instalar ningún programa en nuestro dispositivo; también admite compartir contenido con otros usuarios y permitirles editar nuestro código, de allí su nombre "*Colaboratory*", pues es una herramienta colaborativa que facilita el trabajo en equipo apoyándose en el servicio de almacenamiento de Google Drive. [18]

1.7. TensorFlow

TensorFlow es la principal biblioteca de código abierto para enseñarnos a desarrollar y entrenar modelos de aprendizaje automático. Fue desarrollado por Google para disponer de herramientas con el fin de optimizar las redes neuronales de aprendizaje profundo. Su arquitectura funciona en base a flujo de valores numéricos dispuestos en arreglos multidimensionales conocidos como Tensores, de allí su nombre. Utiliza operaciones matemáticas conectadas a conjuntos de datos para identificar patrones y correlaciones.

También existen algunas otras librerías muy reconocidas como *Keras, Caffe, PyTorch*, *Theano y Scikit Learn*, aunque *Keras* ya se encuentra incluida en el paquete de *TensorFlow* en las últimas versiones. Un aspecto muy importante de *TensorFlow* es que se considera la plataforma de aprendizaje profundo más importante del mundo, cuenta con una gran comunidad de desarrolladores dispuestos a colaborar y resolver dudas. En la actualidad, CocaCola, Twitter, General Electrics, Airbnb, Intel, entre otros, usan esta librería para desarrollar diferentes programas de inteligencia artificial. [19]

Capítulo 2 - Objetivos

2.1 General

Generar un algoritmo de detección y evaluación de actividad física en base a la posición de personas en tiempo real con base en diversas técnicas de inteligencia artificial, que permita el monitoreo en ambientes cerrados y/o abiertos, así como el manejo de grupos de personas o multitudes.

2.2 Específicos

Objetivos específicos del proyecto.

a) Identificar y monitorear la actividad física cuando el paciente está caminando
b) Cuantificar el gasto calórico de las personas en el video
c) Realizar el seguimiento de trayectorias de las personas dentro del entorno real
d) Generar una herramienta que nos ayude a identificar cambios de velocidad, inmovilidad y abandono del lugar.
e) Realizar pruebas con videos aleatorios en diferentes situaciones y ambientes
f) Evaluar la confiabilidad y eficiencia del sistema mediante la comparación de datos manuales o de referencia propio de la base de datos EPFL

Capítulo 3 - Estado del arte

En el siguiente capítulo se muestra de forma literaria la evolución que ha tenido la inteligencia artificial, los antecedentes y/o proyectos actualmente en ejecución sobre inteligencia artificial especialmente en torno a los algoritmos de detección y seguimiento de objetos, para comparar y dar un punto de partida al algoritmo que describe este trabajo. De forma que se comprendan algunos métodos empleados y se pueda tener un punto de vista enriquecido de información actualizada para fortalecer un criterio propio, crítico y constructivo sobre la elaboración del trabajo dirigido.

3.1 Método de investigación o revisión literaria

Para la elaboración del apartado referente a el estado del arte, específicamente en el campo de detección y seguimiento de personas se formula una serie de procedimientos que llevan a obtener artículos, informes, revistas y libros publicados de carácter investigativo, es decir, fuentes confiables de información que se consultaron de diferentes bases de datos o bibliotecas virtuales como por ejemplo el CRAI, IEEE, *Google Scholar* y Scielo. Para optimizar y filtrar la búsqueda se emplean diferentes ecuaciones de búsqueda en dichas plataformas, las cuales están centradas en la pregunta de investigación *¿Como se realiza el seguimiento y detección de personas por análisis de videos?,* para obtener finalmente métodos, algoritmos y técnicas de análisis útiles para el trabajo dirigido.

De aquí se obtuvo mayormente revistas, publicaciones académicas, informes y conferencias de algunos países líderes en inteligencia artificial, como es el caso de Estados unidos, Canadá, China, Taiwán, entre otros.
Esta metodología, junto con demás revisiones literarias, blogs e inclusive contenido digital llevó al entendimiento de los siguientes métodos relacionados a detección y seguimiento de personas. En caso de requerir información adicional se recomienda dirigirse a la bibliografía correspondiente al tema de interés, pues esta información se encuentra redactada para el año 2021, semestre A.

3.1 Algunos métodos para seguimiento de pacientes

Actualmente para el seguimiento de pacientes y monitorización en centros de salud, los métodos más comunes utilizan dispositivos de rastreo a larga distancia y tecnologías activas o pasivas, donde las activas representan dispositivos electrónicos sobre el paciente que consumen energía y necesitan cambiarse después de la descargar o desgastar la vida útil de las baterías (utilizar el teléfono móvil para identificar pacientes), por otra parte tenemos dispositivos pasivos que no consumen energía pero que igualmente son colocados al paciente , por ejemplo, pulseras metálicas con GPS o NFC. [8]

Estos dispositivos utilizan sensores y dispositivos sobre el paciente, lo cual resulta un poco incomodo o invasivo, estos artefactos a su vez pueden perderse relacionado a la condición del paciente, ya que algunas condiciones neurodegenerativas producen perdida del lenguaje, desorientación, cambios de humor, pérdida de motivación, no manejar su autocuidado, entre otros. a continuación, se presentan algunos de los más utilizados:

- **RFID**

  Identificación con radiofrecuencia o RFID por sus siglas en ingles "*Radio-frequency identification*", este sistema, comúnmente usado en lugares como tiendas y supermercados, utiliza un sistema de ondas electromagnéticas generadas por un sensor, para ello necesita de una etiqueta o tarjeta metálica que reaccione a este campo electromagnético y produzca ondas de radio, que son detectadas por un sistema de antenas de radio, encargadas de arrojar una alarma, es así como se evitan robos de prendas de vestir y productos en general. Si bien este método es común y funcional, necesitaríamos que el paciente tenga siempre una etiqueta metálica amarrada a su cuerpo, además de que funcionan como sensores de distancia, por ello no arrojan un dato exacto de posición, solamente de proximidad. [20]

- **BLE**

  Bluetooth de baja energía o BLE (*Bluetooth Low Energy*), es similar al bluetooth clásico, solamente que busca reducir el consumo energético, es una tecnología de comunicación inalámbrica que permite dar una monitorización sobre pacientes, transferencias de información, como por ejemplo música o archivos en dispositivos móviles, parlantes, TV y muchos más. Similar a RFID utiliza ondas de radiofrecuencia, pero en este caso de corto alcance o de baja intensidad en la señal, además de que necesita comunicación en dos sentidos a una frecuencia de aproximadamente 2.4 GHz, por ende, dos dispositivos diferentes, uno sobre el paciente y un receptor. [21]

- **NFC**

  Comunicación de campo cercano o NFC (*Near-field communication*), igualmente basado en RFID, es otro estándar de comunicación entre dispositivos que utiliza ondas de radiofrecuencia para permitir la transferencia de datos, entre sus diferencias esta su ancho de banda de 13,56 MHz y su empleo de 2 antenas por dispositivo, por lo tanto, posee 2 configuraciones diferentes, donde ambos dispositivos transmiten datos, o donde un dispositivo transmite y el otro recibe solamente. Este método es más comúnmente utilizado en tarjetas bancarias, tarjetas de transporte publico recargables y teléfonos celulares, otra gran diferencia se ve en cuanto a intensidad de la señal, pues en NFC son señales débiles de un alcance de apenas 20 centímetros como máximo, por ello se usa bastante en transacciones o pagos electrónicos, pues no permite comunicación a larga distancia. [22]

- **GPS**

  Sistema de posicionamiento global o GPS (*Global Positioning System*), es un sistema que permite localizar objetos sobre la tierra, con una precisión que va desde centímetros hasta metros dependiendo del método especifico, en este método se necesitan igualmente de ondas de radiofrecuencia y permite conocer la ubicación del sujeto en tiempo real, esto se realiza gracias a una red de satélites dispuesta por Estados unidos, de ella se necesitan mínimo 4 satélites, se mide la distancia a cada satélite y se usa un método conocido como trilateración, permitiendo dar una percepción del espacio 3D y obtener así nuestra localización. Este método tiene gran precisión, pero necesita de un dispositivo activo sobre el paciente, puede ser un módulo GPS o un celular. [23] [8]

3.2 Tipos de Redes neuronales artificiales

En relación con el algoritmo de detección y seguimiento de personas, y respondiendo a nuestra pregunta de investigación, *¿Como se realiza el seguimiento y detección de personas por análisis de videos?* Tenemos en este capítulo, una serie de métodos y algoritmos basados en IA, que han sido de gran utilidad en diferentes sistemas de visión por computadora, desde detección de rostros, personas o transeúntes, autos y sistemas de conducción autónoma, que pueden ser útiles en nuestro enfoque principal, para la cuantificación del desplazamiento de sujetos con datos de videovigilancia, algo común en estos métodos es el uso de redes neuronales, pues estas redes poseen gran desempeño para resolver problemas complejos como se explica en todo este capítulo, así como también son tecnologías pasivas no invasivas que pueden dar buenos resultados.

Cuando hablamos de redes neuronales artificiales implica uno de los temas de mayor tendencia y crecimiento respecto a inteligencia artificial puntualmente hablando de *Deep learning* o aprendizaje profundo, estas redes neuronales son sistemas complejos que aprenden de gran cantidad de datos y de su entorno, pero en esta sección se encuentran descripciones para poder diferenciar entre algunos tipos de redes neuronales artificiales.

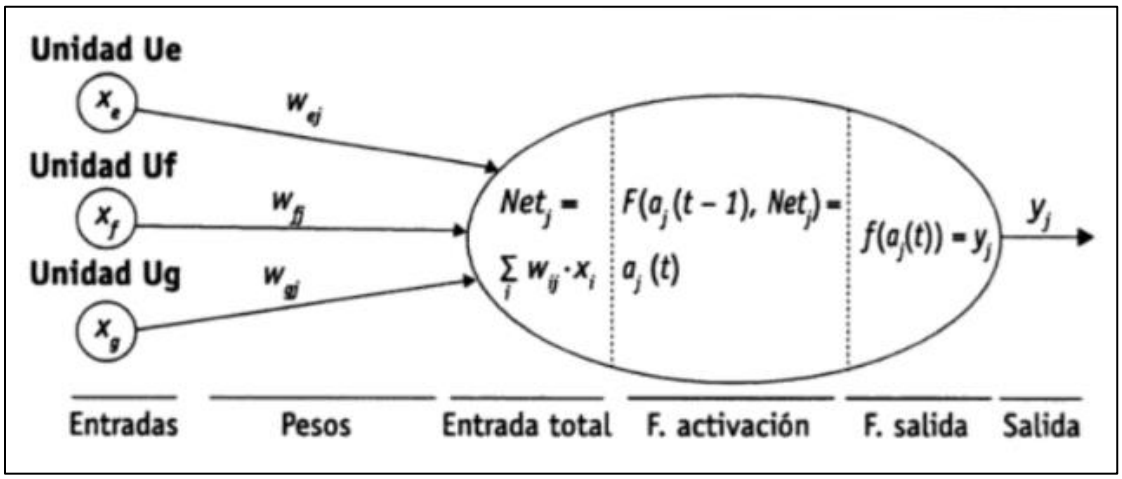

Figura 2. Esquemático de una neurona artificial o nodo. [14]

De manera más detallada (véase Figura 2), el funcionamiento de la neurona se basa en funciones matemáticas encargadas de recibir los valores numéricos correspondientes a una característica de la data o a la salida de la neurona en la capa anterior (X_e, X_f …) luego cada entrada se multiplica por un peso para cada conexión (W_{ej}, w_{fj} …) y se suman estos valores de la multiplicación, este resultado se envía a una función de activación, desde donde se decide si se envía o no información a las siguientes neuronas, teniendo en consideración el valor del *bias* o sesgo $a_j(t)$ que indica que tan sensible es la neurona a la activación. [14]

El entrenamiento de este tipo de redes neuronales se realiza mediante ajustes en las ecuaciones matemáticas para ponderación o peso de las entradas en cada neurona, cuyo ajuste se efectúa en varias iteraciones, este reajuste se realiza basándonos en el error

cuadrático medio entre las salidas y lo que debería ser la predicción correcta, apoyándose también en el algoritmo de gradiente descendiente, donde se busca reducir el error hasta un valor cercano a cero, este algoritmo utiliza un arreglo tridimensional para relacionar los pesos W y el valor X de entrada en nuestra neurona con el desempeño a la salida en función del error. Lo que hace el algoritmo de gradiente descendiente es cambiar los valores de W y X en cada iteración para ir minimizando el error, llegando hasta el mínimo global de la función de error. (Véase Figura 3)

Tal como se menciona, el gradiente descendiente altera los pesos de nuestra red neuronal con cada iteración, pero esto ocurre a una determinada tasa de aprendizaje, la cual indica el tamaño de los pasos que realiza la función de optimización en la parábola de la función de costo con el objetivo de llegar al mínimo (Véase Figura 3), con lo cual, si los pasos son cortos el programa posiblemente llegue a una mejor aproximación de nuestro punto de convergencia, pero esto a su vez toma una gran cantidad de iteraciones, es decir, mayor gasto computacional y bastante tiempo para finalizar el entrenamiento, por otro lado con un paso más largo se logra reducir el número de iteraciones y tiempo, pero posiblemente nunca alcancemos el mínimo pues quedaría oscilando bastante lejos del mismo. Por tal motivo se utiliza comúnmente la decadencia de tasa de aprendizaje, la cual es más adaptable a los cambios de la parábola, pues cuando más alto o empinado sea el gradiente (pendiente de la curva) más alta es la tasa de aprendizaje, en cambio cuando el gradiente sea pequeño más pequeño serán los pasos para llegar al punto de convergencia. [13] [14] [24]

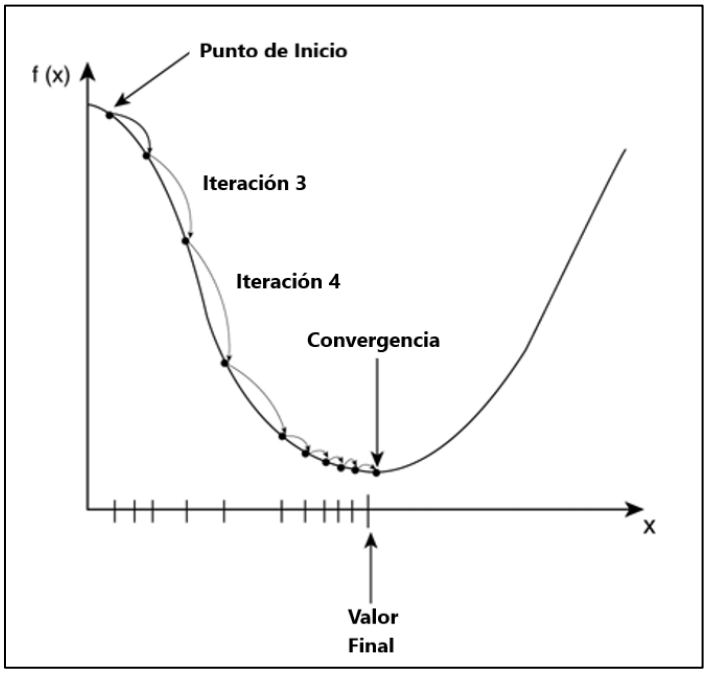

Figura 3. Gradiente descendiente con decadencia de tasa de aprendizaje. [24]

3.2.1 Redes Neuronales Convolucionales (CNN)

Su uso más común es en el procesamiento de imágenes, tiene la misma estructura de una RNA común, con capa de entrada, capas ocultas y capa de salida, a diferencia que en

las capas ocultas se encuentran capas de convoluciones y *MaxPooling*, procesos con los cuales se reduce el tamaño de la imagen a medida avanza desde la capa de entrada hacia las capas ocultas, así a medida se acerca a la capa de salida la información es más abstracta. [25]

De forma resumida, las CNN poseen 4 componentes principales:

1. Una capa de entrada que contiene los valores de los pixeles de la imagen

2. Una capa convolucional, la cual reduce a componentes principales como se explica más adelante, directamente después esta la Unidad lineal rectificada o ReLu, la cual contiene una función de activación que convierte todos los valores negativos del mapa de características en cero y ayuda a introducir no linealidad para mejorar las predicciones del mundo real (datos no lineales), esta función de activación es bastante común en las CNN y permite mostrar la salida de la capa convolucional.

3. Una capa de *pooling* o agrupación, que reduce el número de parámetros o dimensionalidad de la imagen resultante de la capa anterior.

4. Las capas completamente conectadas, aquí se producen las operaciones normales de una red neuronal, para asignar puntajes y probabilidades desde sus funciones de activación, dichos resultados determinarán finalmente la decisión o resultado de la predicción. Igualmente se recomienda usar Softmax como función de activación en la capa de salida para mejorar el desempeño. [25]

Para su entrenamiento se utiliza un método conocido como *back-propagation* o propagación hacia atrás, el cual busca optimizar los pesos y *bias* dentro de nuestra CNN. Su nombre se debe a que esta optimización se realiza desde las capas finales o de salida hacia la capa de entrada, es decir de atrás hacia delante. Aquí básicamente, se inicializan los *bias* en cero y los pesos según una distribución normal de conveniencia, posterior a ello se comienza a ingresar información en la CNN, se procesa la información de neurona a neurona con estos pesos y se produce una predicción en la capa de salida. Esta predicción al no ser optimizados los pesos y *bias* de la CNN deben tener un error alto, aquí interviene la gráfica de gradiente descendiente, entonces, usando fórmulas matemáticas de derivadas parciales que surgen de evaluar como varia la salida con respecto a cada variable, se van modificando dichos pesos y *bias* para que el error sea cercano a cero.

Las funciones de activación como ReLu, Softmax, Softplus, sigmoide, TanH y demás, emplean ecuaciones matemáticas que representan o modifican la data de entrada para producir las respectivas salidas. Pueden emplearse diferentes funciones de activación para diferentes capas de nuestra red neuronal, siendo de las más empleadas en la capa de entrada la función Sigmoide (Véase Figura 4), pues convierte valor de la entrada a un rango entre 0 y 1, facilitando cálculos matemáticos de capas posteriores, también en las capas ocultas se puede usar Softplus o ReLu, ya que ayudan a configurar los pesos y *bias* de la red neuronal durante el entrenamiento para evitar la linealidad dando flexibilidad al programa y evitando sobreajuste, finalmente en la capa de salida se puede usar la función de activación de Softmax, con la cual, en la salida de nuestra red neuronal tendremos valores de probabilidad en base a todas los valores de la capa anterior dada su función. [26] [27]

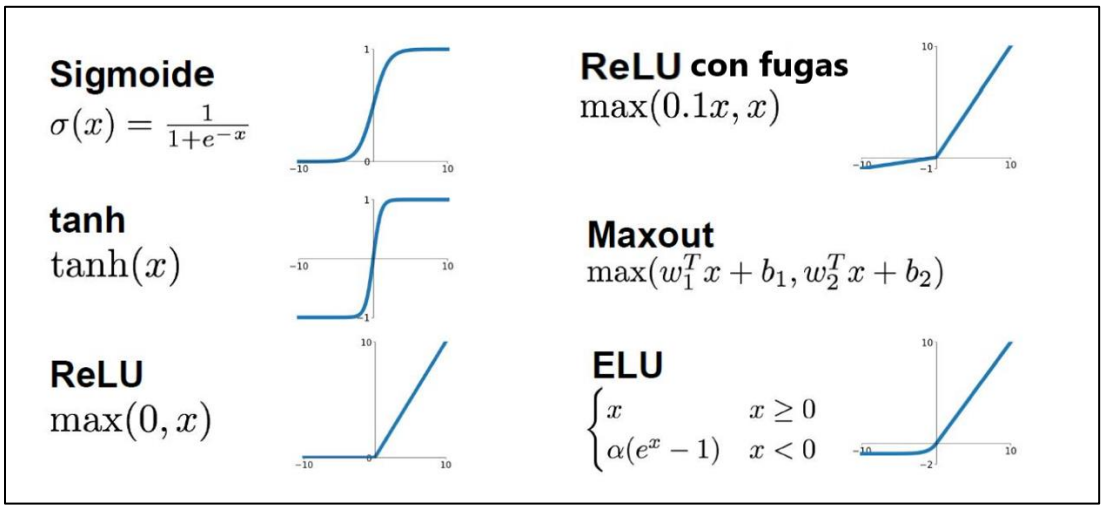

Figura 4. Gráfico principales funciones de activación. [27]

Además, como se observa (Véase Figura 5), el resultado final de la convolución posee menor tamaño comparado con la imagen original, esto es básicamente lo que realiza la convolución, pues traslapa dos funciones, en este caso dos matrices, para llegar a un resultado que contenga la información de cada parte, es decir, no se pierde información, simplemente se transforma con el objetivo de reducir el gasto computacional necesario para procesar la imagen. Entre más convoluciones tenga el algoritmo, aumenta también el número de características importantes extraídas de la imagen, la profundidad de la imagen y por ende mejora el desempeño de nuestra red neuronal para identificar patrones entre la data. [28]

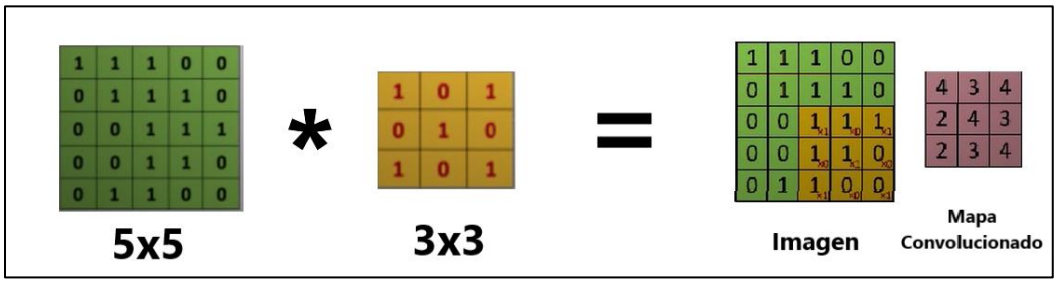

Figura 5. Convolución de imagen 5x5. [28]

Generalmente posterior a una convolución se encontrará un *MaxPooling o downsampling*, el cual utiliza otro filtro, como una ventana para reducir la dimensionalidad del mapa de características resultante de la convolución, aunque de igual forma retiene la información más significante, lo que realiza esta función es utilizar una ventana para elegir el elemento con el mayor valor dentro de esa ventana para un "*pooling*" o agrupación, similar a la convolución esta ventana se desplaza por todo el mapa de características para finalmente obtener una matriz manejable para el algoritmo, reduciendo los parámetros para mejorar la eficiencia del algoritmo, minimizando el trabajo computacional y evitando así un sobreajuste en nuestro entrenamiento.(Véase Figura 6) [25]

Por ejemplo, en la primera capa de convolución y *Maxpooling* va a identificar líneas horizontales, verticales, diagonales, la siguiente capa identifica formas como círculos,

cuadrados, etc. Y en la siguiente capa se identifican elementos más elaborados como rines, puertas, llantas, etc. Básicamente con estas funciones de convolución y *Maxpooling* permitimos que el algoritmo trabaje de forma eficiente y rápida.

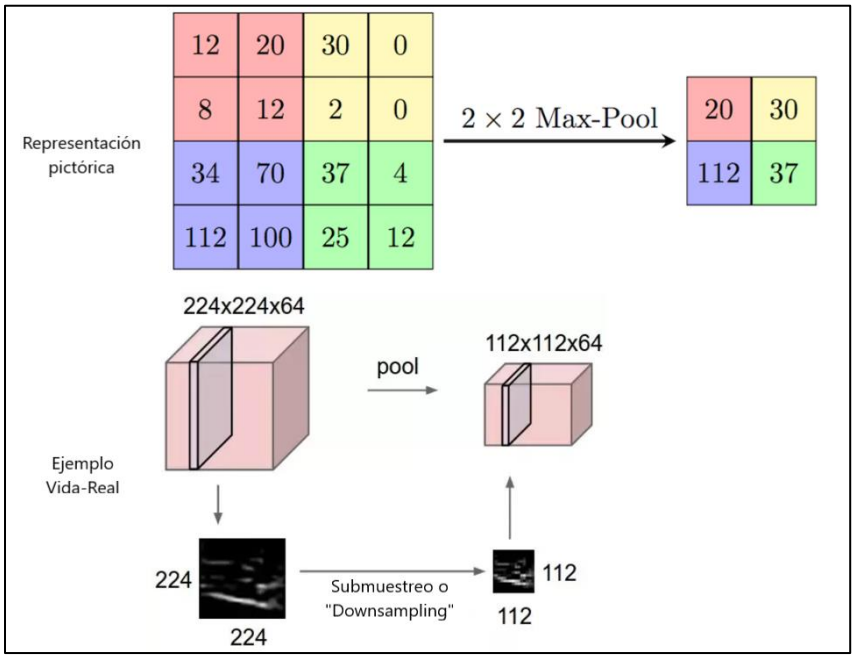

Figura 6. Representación de Maxpooling. [29]

En la Figura 6 podemos apreciar como en la capa de *Maxpooling* no se cambia la dimensionalidad de la información, como ejemplo en la imagen a blanco y negro, que corresponde al resultado de una capa de convolución se tienen 224x224 pixeles y profundidad de 64, posterior al *Maxpooling* se obtiene 112x112 pixeles, pero la profundidad sigue siendo la misma. Contrario a lo que sucede en la capa de convolución, pues la idea es aplicar distintos kernel o filtros para la misma imagen, de forma que disminuye su número de pixeles, pero aumenta en profundidad, pues tendríamos no solo las 3 dimensiones de RGB, sino que tendríamos 3•n dimensiones, correspondientes a los n filtros o convoluciones que se apliquen a la imagen. [29]

Existen también redes que utilizan *average pooling*, donde no se elige el valor máximo de la ventana o filtro, sino que se calcula el valor promedio en la ventana y ese valor es el que se usa en el mapa de características. En general, las arquitecturas CNN son pioneras y ampliamente utilizadas en clasificación de objetos, donde le damos una imagen a la entrada y el algoritmo nos arroja una predicción del objeto que contiene la imagen, pero no su ubicación. Actualmente este tipo de red neuronal ha evolucionado a métodos de detección, los cuales pueden clasificar varios objetos y dar su posición en la imagen, basado en esta estructura tenemos métodos como *Fast R-CNN y Mark R-CNN*. [30] [31]

3.2.1.1 Redes convolucionales basadas en región (R-CNN), 2014

Usando solo las CNN tenemos algoritmos de clasificación, donde podemos identificar si una imagen corresponde a un objeto, pues a partir de su mapa de características

principales gracias a las redes convolucionales tenemos un resultado en porcentaje de que una imagen es por ejemplo un 70% un perro, pero ¿qué pasa si queremos detectar varios objetos en una misma imagen? ¿O si queremos conocer la posición del objeto dentro de la imagen?, ya que con las redes convolucionales estamos analizando toda la imagen en búsqueda de características de dicha imagen, por ello, lo que proponen las R-CNN es un modelo o algoritmo de detección, en el cual, se tenga un análisis por regiones y no de toda la imagen en general, lo que buscamos con estos algoritmos es generar un cuadro delimitador, el cual contenga el objeto dentro de la imagen.

Lo que realiza el R-CNN es proponer un gran número de cuadros delimitadores o regiones en nuestra imagen para revisar si alguno de estos cuadros corresponde con un objeto, estos cuadros se obtienen de un proceso conocido como búsqueda selectiva de alto nivel (Véase Figura 7), este proceso crea diferentes ventanas en la imagen y otorga alta importancia a aquellas ventanas con características similares de color, intensidad o textura entre sus pixeles, así mismo, se comparan estas propias ventanas en busca de similitudes y se combinan para generar una ventana más general del posible objeto, posteriormente la ventana ingresa a una red convolucional previamente entrenada, donde se extraen las características de cada región propuesta, determina si las regiones pertenecen a algún objeto en particular mediante un sistema de clasificación con SVM (*Support vector machine*, "Máquina de vectores de Soporte"), el cual básicamente es un clasificador que busca optimizar la región intermedia entre clases, para reducir el error en la clasificación debido a datos similares a dos clases diferentes. Este proceso se repite para cada cuadro delimitador y adicional a ello, el algoritmo ajusta el cuadro delimitador a cada detección mediante un "*Bounding Box regressor*" o regresor del cuadro delimitador, el cual es un algoritmo que intenta incluir todos los pixeles del objeto en un mismo cuadro para así ajustar el cuadro delimitador a la detección, según su entrenamiento previo por métodos como regresión lineal. [32]

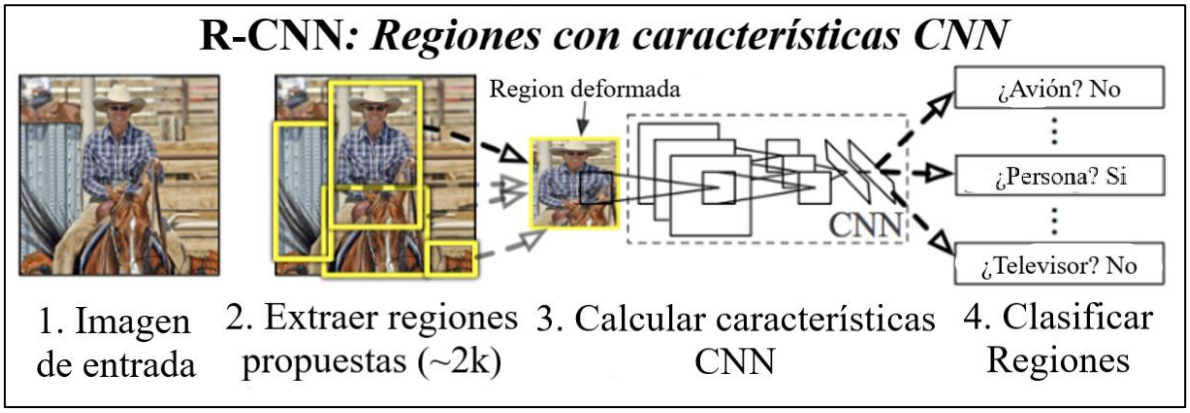

Figura 7. Arquitectura R-CNN. [33]

Con este modelo finalmente tenemos la detección de cada objeto dentro de la imagen junto con su cuadro delimitador, y la información de la clase, es decir, a que objeto pertenece. En resumen (Ver Figura 7), se extraen regiones de interés de la imagen de entrada (paso 2), se calculan las convoluciones para obtener el mapa de características de cada región (paso 3) para finalmente clasificarlo como un objeto (paso 4) y así obtener la imagen con el cuadro delimitador y su clase.

Una de las características de la R-CNN es que su ejecución para la detección de imágenes es bastante demorada, pues usualmente se utilizan cerca de 2000 regiones propuestas o cuadros delimitadores, las que finalmente pasan a la extracción de características para su posterior clasificación, por ende, toma cerca de 49 segundos analizar una sola imagen. Respecto a su entrenamiento requiere también de bastante tiempo, pues consta de 3 modelos diferentes, la CNN para extraer mapa de características, el SVM para clasificación y el regresor lineal para ajustar el cuadro delimitador de la detección. [33]

3.2.1.2 Redes convolucionales basadas en región rápida o Fast R-CNN (2015)

Este tipo de redes neuronales igualmente utiliza la búsqueda selectiva de alto nivel para obtener las regiones propuestas, pero adicionalmente, agrega el módulo de agrupación de regiones de interés (*RoI pooling*), modificando el orden, con lo cual, cada región propuesta ya no se envía a una CNN para extraer su mapa de características (Ver Figura 7), en cambio, se envía directamente toda la imagen de entrada a la CNN para obtener un mapa de las características principales y este es el que se separa por regiones de interés predefinidas (*RoI projection*), luego es deformado hasta llegar a un tamaño correcto por una capa de *RoI pooling*, del cual se obtiene un vector de características para cada región de interés, cada uno de estos vectores o mapas de características reducido pasan a una capa Softmax donde se realiza la clasificación, y también a un algoritmo Regresor del cuadro delimitador para ajustar los cuadros limitadores en cada región de interés.(Véase Figura 8)

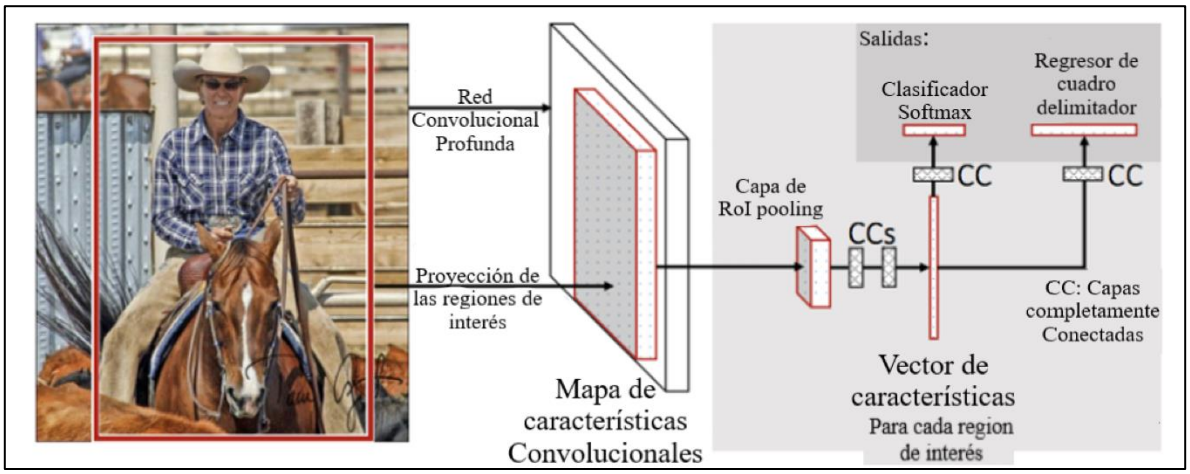

Figura 8. Arquitectura Fast R-CNN. [34]

Fue creada por los mismos desarrolladores de la R-CNN, pero con algunas modificaciones para mejorar su velocidad de procesamiento, a esto debe su nombre, ya que no solo se logró reducir el tiempo de entrenamiento necesario, sino también el tiempo de detección en una imagen, pasando de 49 segundos a solo 2,3 segundos, un cambio abismal. [33] [34]

3.2.1.3 Redes convolucionales basadas en región más rápida o Faster R-CNN (2015)

Los algoritmos anteriores usaban la búsqueda selectiva para determinar las regiones de interés propuestas, pero este proceso es bastante lento y afecta el rendimiento de la red, por ello se creó la *Faster R-CNN*, este método elimina la búsqueda selectiva y en su lugar emplea una red neuronal (RPN o *Region Proposal Network*) para determinar las regiones

de interés, aquí lo que sucede es que se pasa una ventana a el mapa de características de la imagen que llega desde la CNN y de cada ventana genera unos cuadros delimitadores potenciales junto con puntajes de que tan buenos cuadros pueden ser para cada anchor box o caja de anclaje, estas cajas son unos cuadros que dividen la imagen similar a colocar una cuadricula de 4x4 divisiones sobre la imagen, para dar origen a un número menor de regiones propuestas comparado con la búsqueda selectiva, por ende, menos gasto computacional. Luego estas regiones igualmente pasan a un *RoI pooling*, donde se proyectan sobre el mapa de características de la imagen de entrada (igual a el *Fast R-CNN*) para obtener un vector *RoI* con la información para clasificar cada región de interés con Softmax y obtener su cuadro delimitador con un algoritmo de regresión lineal.

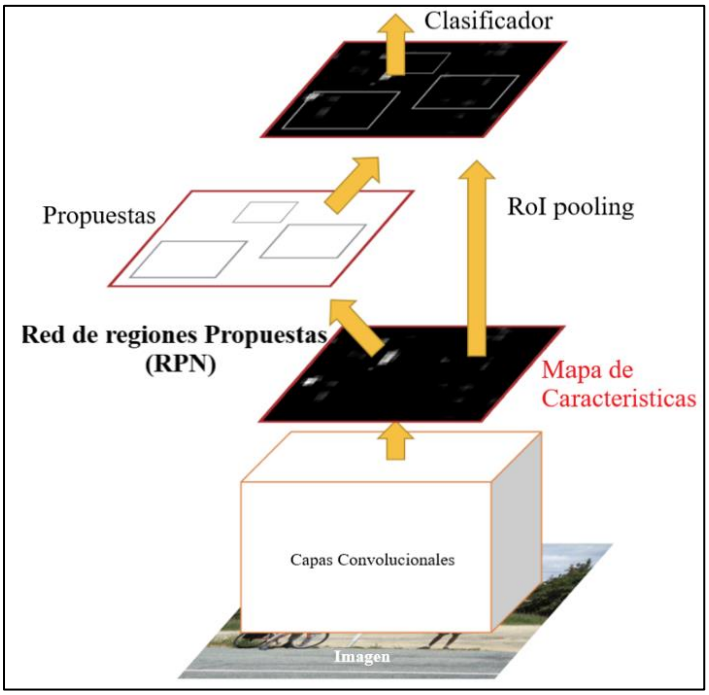

Figura 9. Arquitectura del Faster R-CNN. [33]

Igualmente, este método de Faster R-CNN es bastante más veloz que su predecesor, el algoritmo Fast R-CNN, ya que, en este último, la detección de objetos en una imagen tardaba 2,3 segundo, en cambio con este "nuevo" método solo le toma 0,3 segundos analizar una imagen. [33]

3.2.1.4 Mask R-CNN (2017)

Este método es una extensión de Faster R-CNN, con los algoritmos anteriores obtenemos un cuadro delimitador y la clase, pero con la arquitectura de Mask R-CNN también tenemos la máscara del objeto, es decir, su contorno píxel por píxel. Este algoritmo tiene ciertas diferencias, ya que los autores agregaron una rama adicional paralela al Softmax y el regresor lineal, esta es una red convolucional encargada de la predicción de máscaras para cada objeto, que arroja como resultado una máscara binaria, con valor de 1 para los pixeles que pertenecen al objeto y valor 0 para los pixeles que no, además, el Mask R-CNN no posee una capa de *RoI pooling*, si no una capa de *RoI Align* o alineación de las regiones de interés, ya que según varios autores, las regiones propuestas en los mapas de

características del *RoI pooling* no se encontraban alineados con las regiones propuestas en la imagen original, es decir, que el cuadro delimitador de una predicción podría encontrarse desalineado con respecto a la imagen original, aunque a simple vista no se observa, cuando hablamos de segmentación pixel a pixel puede llegar a producir errores notables.(Vease Figura 10). [35]

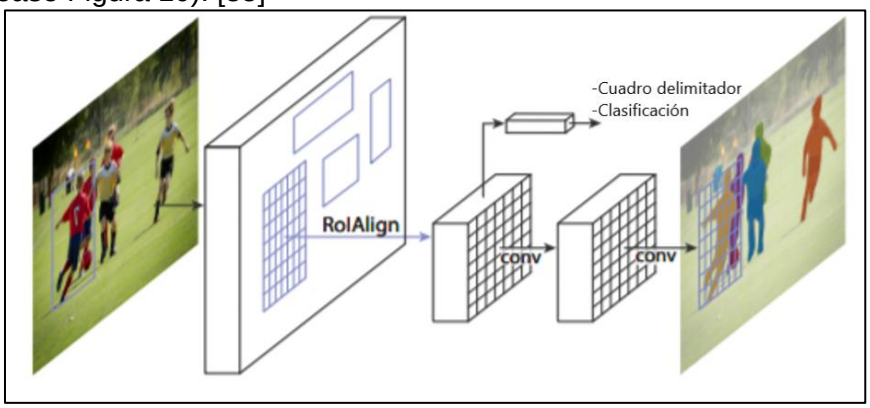

Figura 10. Arquitectura de Mask R-CNN para segmentación. [35]

En algoritmos pasados, con CNN simples como clasificador obteníamos solamente las clases que se encontraban en la imagen de entrada (Figura 11, a), posterior a ello con el algoritmo de detección R-CNN o Fast R-CNN obtenemos un cuadro delimitador para cada clase o etiqueta (Figura 11, b), esto gracias al algoritmo de búsqueda selectiva, en donde se obtenía una segmentación semántica de la imagen sin diferenciar las instancias de un mismo objeto (Figura 11, c), pero en el algoritmo de Mask R-CNN se da segmentación de instancias (Figura 11, d), donde calcula una máscara para cada objeto incluso si son de la misma clase, de allí su nombre. [36]

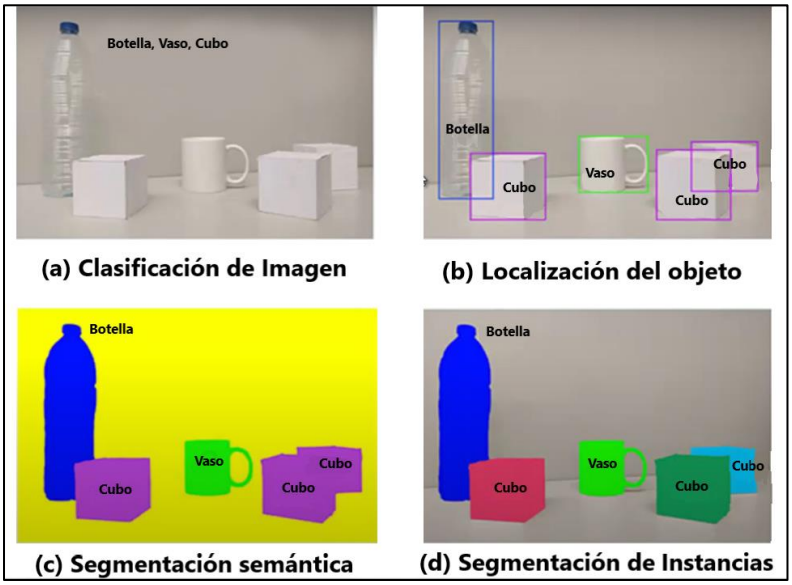

Figura 11. Ejemplos de segmentación y resultados en la detección de imágenes para diferentes tipos de R-CNN. [37]

3.3 Yolov1

Este algoritmo similar a los vistos como R-CNN es un detector de objetos en imágenes, con lo cual, también nos arroja como resultado un cuadro delimitador y la predicción para el objeto en dicho cuadro. Una de las diferencias con los algoritmos anteriores son sus etapas, ya que por ejemplo el Faster R-CNN contiene dos redes neuronales, una para obtener regiones de interés y otra para clasificar los objetos, y YOLO solamente necesita de una red neuronal para todo el trabajo. YOLO Versión 1 se entrenó con la base de datos PASCAL VOC, que contiene imágenes correspondientes a 20 clases diferentes. Este algoritmo fue creado en 2005 por *Joseph Redmon* junto con el grupo de investigación de inteligencia artificial de Facebook.

Algunas características de YOLO es que aumenta considerablemente su velocidad de procesamiento, logrando una detección de aproximadamente 45 fotogramas por segundo, o en su versión Fast YOLO una detección de hasta 155 fotogramas por segundo, lo cual es bastante impresionante comparado con otros métodos basados en CNN, esto lo logra al utilizar una sola red convolucional para realizar la clasificación y generar los cuadros delimitadores, su arquitectura posee 24 capas de convolución y como entrada necesita una imagen de entrada de 448x448 pixeles (Véase Figura 12). Igualmente, YOLO logra un rendimiento superior a otros detectores, ya que muestra una precisión por encima del promedio, esto a una velocidad que puede ser empleada en tareas en tiempo real.

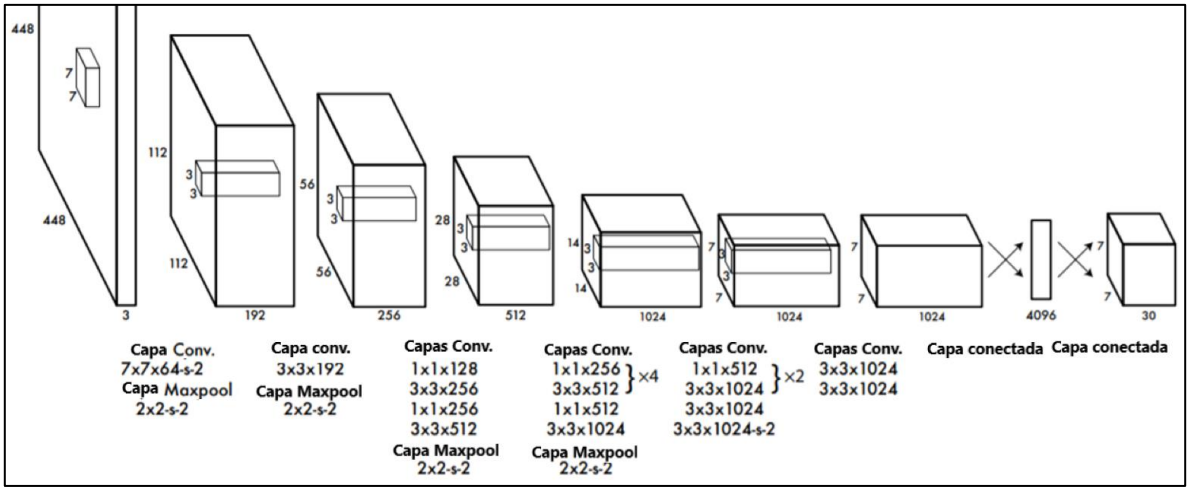

Figura 12. Arquitectura de YOLO v1. [38]

Otra característica de su arquitectura es que está basado en el modelo de Darknet, en parte mantiene su estructura y pesos, solamente agregando algunas capas adicionales y optimizaciones específicas, esto se conoce como *Transfer learning*, ya que estamos usando un modelo preentrenado en una estructura más grande con un enfoque diferente, es decir, se transfiere el conocimiento de *Darknet* a nuestra CNN, lo cual ha demostrado dar mejores resultados de rendimiento en algunos algoritmos y asimismo logra reducir el tiempo necesario para entrenar a YOLO. Solamente se le agregan 4 capas convolucionales y 2 capas completamente conectadas al modelo *Darknet*. [38]
En su funcionamiento, se diferencia de otros algoritmos ya que como lo dice su nombre "*You Only Look Once*" (YOLO), en español "solo miras una vez" simplemente repasa la imagen una vez, es decir, utiliza toda la imagen como entrada y no necesita crear regiones

de interés y analizar varias veces una misma región, eliminar duplicados de un objeto, entre otros procesos que toman bastante tiempo y que se usaban comúnmente en los algoritmos en base a regiones y clasificadores como R-CNN.
Esto lo logra viendo el problema como una regresión lineal y dividiendo la imagen por celdas, según el paper original en una imagen de 7x7 cuadriculas (aunque esta división puede cambiar), cada cuadro de división o celda conocido como anchor o ancla se encarga de generar B cuadros delimitadores para posibles objetos dentro de dicha celda, estos cuadros delimitan un mismo objeto, de forma que debemos elegir solamente uno, aquí es donde usamos la *IoU o intersection over union*. Esta básicamente es una relación entre el área del cuadro delimitador de la predicción y el cuadro delimitador verdadero (en el caso del entrenamiento del algoritmo), una vez entrenado, para las predicciones se comparan los múltiples cuadros delimitadores B con el cuadro delimitador de mayor probabilidad o confianza, en la Figura 13 es aquel con probabilidad de 0,9, la IoU calcula (área de intersección/el área de unión) entre este cuadro azul con el cuadro delimitador de probabilidad 0,6 y según un umbral, en ese caso si el umbral es por ejemplo 0,5 y el IOU es 0,51 entonces al ser mayor esta IOU se elimina el cuadro de 0,6, este método se conoce como *Non max supression* o “Supresión no Máxima” en español, busca eliminar todos aquellos cuadros delimitadores que no sean máximas probabilidades. En forma más simple es elegir el cuadro de mayor probabilidad, que sería aquel del cual nuestra red neuronal está más seguro que es un objeto, pero ¿porque simplemente no se elige este cuadro y ya?, imaginemos que en el caso de la Figura 13 tuviéramos dos automóviles, puede que elimine el cuadro delimitador del otro automóvil, ya que no discrimina entre objetos de una misma clase, en cambio con supresión no máxima al encontrarse sin o con poco contacto entre los cuadros delimitadores de ambos vehículos, el IoU no supera el umbral y, por ende, no elimina el cuadro, es decir, asume que no son el mismo objeto y al final tendríamos un cuadro delimitador para cada auto.

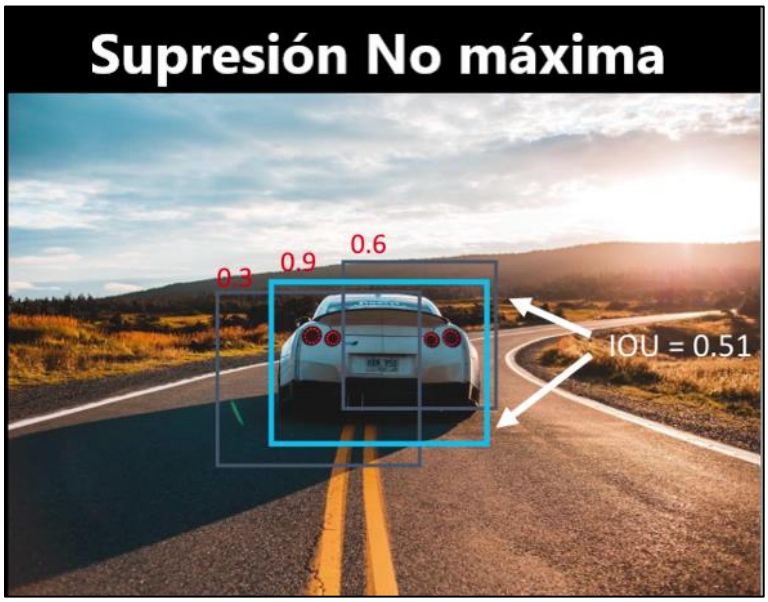

Figura 13. Ejemplo IoU y Supresión no máxima. [39]

En cada celda tenemos 5 datos, una confianza asignada Pc (si encuentra un objeto será 1 y si no entonces ese puntaje es 0), así se ignoran los cuadros que no contengan

información, junto a Pc tenemos la información del cuadro delimitador (Bx,By,Bw,Bh) correspondiente al centro del objeto xy, a la altura h y el ancho w en pixeles. Pueden darse casos donde el objeto se encuentre en varias celdas al mismo tiempo, por ello, el encargado de dar la información es la celda donde se encuentra el centro del objeto. En caso de que se encuentren varios centros de objetos distintos en la misma celda, se debe aumentar la dimensión de este vector, por ejemplo, en la Figura 14 se aumenta al doble debido a la presencia de dos objetos en la misma celda, aquí también podemos ver lo que sería cada vector independiente para el perro y la persona, y lo que tendríamos a la salida de YOLO, el cual es el vector a la derecha del todo. Donde c1 y c2 son las dos clases de objeto que detectaría este ejemplo.
Nota: Este método es usado en Yolo v2 y se conoce como anchor boxes, ya que Yolo v1 solo acepta un objeto por celda.

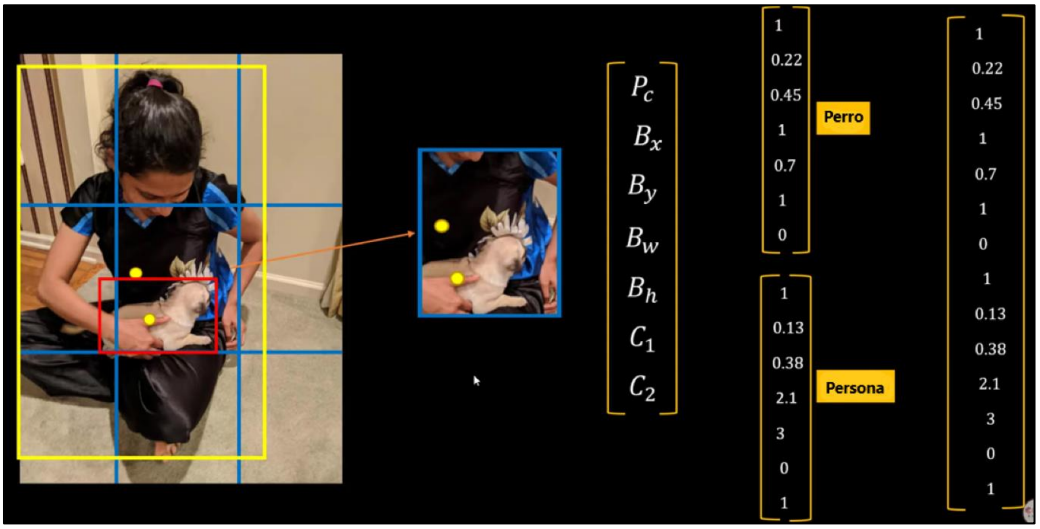

Figura 14. Ejemplo de YOLO con división en celdas 3x3, con dos objetos en una misma celda. [40]

Algunas desventajas de YOLO v1 es que, por su mismo diseño en celdas divididas, le resulta difícil identificar objetos pequeños al interior de una misma celda (normalmente se identifica un solo objeto por celda), esto se podría solucionar aumentando el número de celdas, pero a su vez implicaría mayor gasto computacional. [38]

3.4 Yolov2

Este algoritmo es una mejora de su versión anterior (YOLO v1), fue igualmente diseñado por Joseph Redmon y Ali Farhadi en 2016 bajo el nombre de YOLO 9000, cuyo nombre proviene del hecho que Yolo es capaz de detectar más de 9000 clases y mantener su velocidad en tiempo real, a 67 FPS obtuvo un puntaje de 78,6 mAP (*mean average precision*), cuyo valor sirve de referencia con otros métodos analizados respecto a su precisión en la base de datos VOC 2007, superando a métodos como Faster RCNN con ResNet y SSD (*Single Shot Detector*) en precisión y en velocidad también. En esta ocasión los autores probaron también con otros sets de datos como *COCO dataset e ImageNet*, para detección y clasificación respectivamente.

Respecto a su arquitectura está basado en *Darknet* 19, el cual contiene 19 capas convolucionales y 5 capas de *maxpooling*, además, tenemos varias mejoras que llevaron al rendimiento superior en precisión. [41]

- Anchor boxes o cajas de ancla para predecir los cuadros delimitadores, se removieron las 2 capas completamente conectadas al final de YOLO v1 y se utilizó un sistema de cajas de ancla, que asigna un mismo vector cuando tenemos dos objetos o más en una misma celda (Véase Figura 14), con lo cual se disminuye la precisión, pero se aumenta la recuperación o sensibilidad del algoritmo.
  Igualmente, como en YOLO V1, cada anchor box posee B cuadros delimitadores, por consiguiente, para cada anchor box se elige el mejor cuadro delimitador usando IoU y *non max supression*.

- En YOLO v2, se realizó el entrenamiento desde varias resoluciones, siendo la mínima 320x320 pixeles, y la máxima 608x608. De esta forma, el algoritmo aprende a trabajar con diferentes resoluciones a la entrada. También se observó que a menor resolución YOLO v2 trabaja más rápido en comparación con imágenes de alta resolución.

- En su arquitectura usamos *Darknet* 19, este a su vez dentro de una red neuronal basada en *GoogleNet*, la cual es una red neuronal para clasificación de imágenes, con esta estructura se demostró una mayor velocidad comparado con arquitecturas como VGG-16, aunque posee menor precisión, exactamente un 2% menos preciso. Además, similar a VGG, *Darknet* utiliza filtros 3x3 en algunas capas y *average pooling* al final de su arquitectura.

3.5 Yolov3

Es considerado como estado del arte en redes neuronales para detección de objetos debido a sus mejoras en precisión y velocidad alta comparada con otros métodos, logrando una precisión aceptable a una velocidad bastante superior a los demás métodos, con lo cual posee gran potencial en aplicaciones en tiempo real. Fue diseñado igualmente por Joseph Redmon y Ali Farhadi en 2018.

Respecto a su rendimiento mostró un mejor desempeño en velocidad comparado con métodos de detección como *Retinanet* 50 y *Retinanet* 101, modelos que fueron considerados como los mejores en el año 2017, con unos tiempos más cortos para cada iteración, aunque respecto al *mAP* o precisión no alcanza a su competencia directa, si es un valor aceptable considerando una diferencia de 4 a 1 en velocidad.

Fue elaborado en la arquitectura de *Darknet* 53, la cual contiene 53 capas de convolución y presenta mejor precisión que su versión anterior *Darknet* 19, además, posee otras 53 capas adicionales dedicadas a la detección esta es una de las principales razones por las que es también más lento que YOLO V2, pues posee muchas más capas de procesamiento. Con esto se mejoró la detección de objetos pequeños, se mejoró la localización de los cuadros delimitadores junto con la capacidad de detectar imágenes con diferentes aspectos, radios y escalas. Adicional, se incluyeron bloques residuales, los cuales saltan o

eliminan conexiones entre neuronas como en redes comunes, por defecto, las salidas de una capa de la red neuronal van directamente conectadas a la entrada de la capa siguiente, lo que supone este tipo de estructuras es conectar la salida de una capa a la entrada de 2 o más capas adelante. (Véase Figura 15)

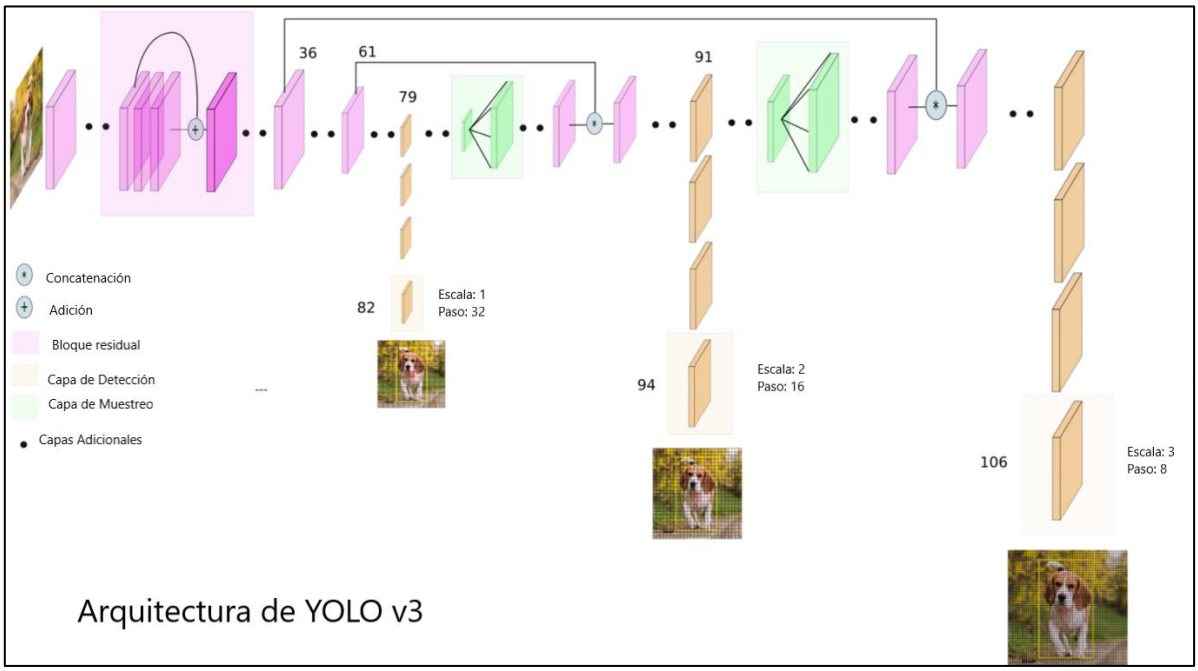

Figura 15. Arquitectura de YOLO v3. [42]

Modificaron la predicción en base a modelos de redes piramidales de características (*Feature pyramid network* o FPN), de aquí se obtiene el mapa de características de 2 capas atrás, se concatenan el mapa de características generales y el mapa pequeño de características posterior a una interpolación o expansión por un factor de 2x usando el método de vecinos cercanos, de forma que se aumenta la información y se obtiene características de alta y baja resolución. Con estos modelos lograron hacer detecciones a 3 diferentes escalas de la imagen como se observa en la Figura 15, es decir, 3 mapas de características diferentes, un mapa de 13x13x255 que detecta objetos grandes (en la capa 82), otro mapa de 26x26x255 que detecta objetos medianos (capa 94) y otro mapa de características de 52x52x255 que detecta objetos pequeños (capa 106), ya que cada núcleo de detección tiene la forma 1 x 1 x (B x (5 + C)), donde el valor 5 es la información x, y, w, h, Pc del cuadro delimitador, B=3 es el número de cuadros delimitadores y C=80 el número de clases. Con estas modificaciones se obtiene mejores predicciones a diferentes tamaños o aspectos de la imagen de entrada. [43] [42]

3.6 Yolov4

Una gran diferencia en este algoritmo son sus autores, puesto que los autores de las versiones anteriores, Joseph Redmon y Ali Farhadi se retiraron del proyecto, Joseph anunció en su cuenta de Twitter que fue debido al uso militar que se le estaba dando a su

tecnología, junto con todas las implicaciones sociales que había observado, en cambio Ali Farhadi fundó una compañía llamada Xnor.ai que fue adquirida por Apple tiempo después. Fue entonces en 2020 cuando surgió YOLO v4, un algoritmo diseñado por Alexey Bochkovskiy, Chien-Yao Wang y Hong-Yuan Mark Liao, que mostró una notable mejora en su desempeño, tanto en velocidad como en precisión comparado con otros métodos como YOLO v3. [44]

Un cambio en YOLO v4 es la "*Data Augmentation*" o aumento de datos, es la interpretación de la misma información desde diferentes puntos de vista, este fue una de las mejoras respecto a YOLO v3 en su entrenamiento, se basa en modificaciones pixel por pixel en las imágenes de entrenamiento, cambios de color, textura, parches negros o blancos, cortes y demás modificaciones sobre la imagen que ayudan al algoritmo a aumentar su precisión y flexibilidad, pero sin afectar su rendimiento en términos de velocidad. Este cambio de entrenamiento los autores lo denominan "*Bag of freebies*", bolsa de regalos en español, pues implica un aumento en el costo de entrenamiento, pero aumentando la precisión sin costos en hardware, de forma que obtenemos un mejor desempeño gratis.

Para su arquitectura se basaron en distintas etapas del algoritmo (Véase Figura 16), que a su vez pueden tener varias subestructuras en su interior, por ello los autores realizaron pruebas para determinar los modelos más adecuados, en primer lugar tenemos la columna vertebral que es la estructura principal para extraer el mapa de características de la imagen, en este caso observaron el mejor rendimiento con CSPDarknet53 para el set de datos de MS COCO, este es una modificación de Darknet53 usado en YOLO v3, que implementa la base del modelo *CSPNet* basado en conexiones parciales cruzadas y permite obtener un mapa de características más significante o elaborado, sin cambiar la velocidad del algoritmo. Para el cuello utilizan *path-aggregation network* o PAN, que similar a la FPN de YOLO v3 busca crear diferentes niveles de mapas de características desde los cuales observar objetos a diferentes escalas. Y finalmente la cabeza, en este caso se mantiene la cabeza de YOLO v3 con algunos ajustes de optimización. [44]

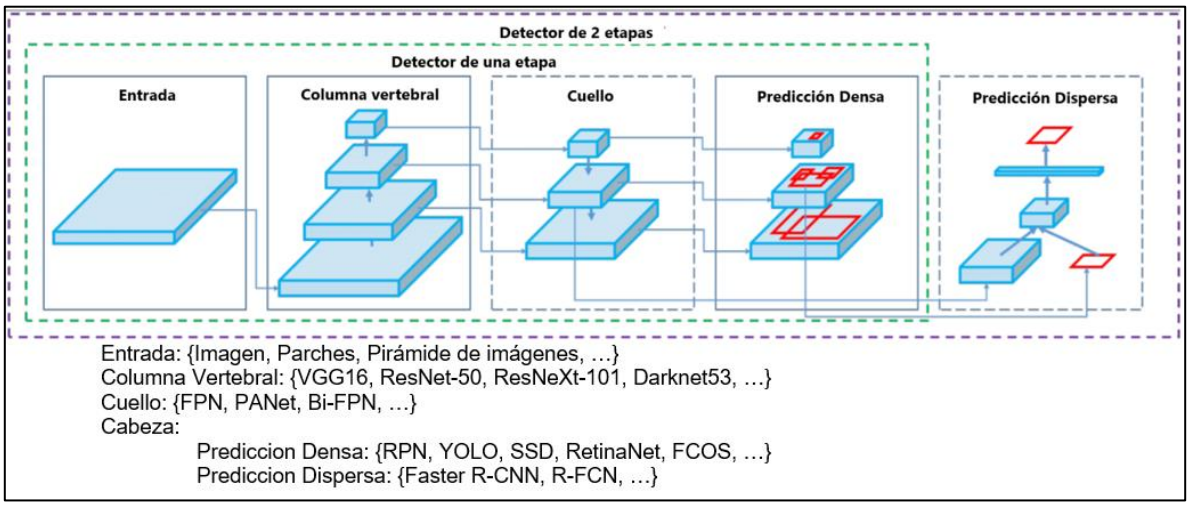

Figura 16. Arquitectura YOLO v4. . [44]

También existen otra gran cantidad de modificaciones, basándose en métodos ya existentes, en las que ellos incluyen la "*bag of specials*" o bolsa de descuentos, similar a la

anterior representa un aumento considerable en la precisión del algoritmo, aunque esta vez sí está relacionado con un costo en velocidad, que es bastante bajo, por ello con los mínimos recursos obtenemos las máximas ganancias, tal como una oferta comercial.

En su entrenamiento se usó el set de datos de *ImageNet y MS COCO dataset*, y debido a la gran cantidad de métodos a probar, se utilizó el método de *Ablation Study*, donde se eliminan y agregan componentes de la inteligencia artificial con el objetivo de observar su influencia en la salida de le Red neuronal, luego se registra la información en una tabla y se puede llegar a una mejor toma de decisiones para definir la arquitectura del modelo.

3.7 Método de cambio medio para seguimiento

Ahora debemos entender la diferencia entre detección y seguimiento, si bien la detección es más común y conocida, el seguimiento se encuentra en auge y es utilizado en aplicaciones múltiples como seguimiento de autos, peatones, balones de futbol en un partido, etc. La detección se caracteriza como hemos visto, en la clasificación de uno o varios objetos en una sola imagen, mostrando su clase y localización dentro de la imagen, por otra parte, el seguimiento realiza varias detecciones para cada imagen del video, con la particularidad de que identifica el objeto y no lo pierde de vista, es decir, cuando hablamos de seguimiento estamos rastreando la localización de uno o varios objetos en particular a lo largo del video, revisando las detecciones de la imagen anterior y comparándolas con las detecciones actuales para asignarle el mismo ID al mismo objeto del fotograma anterior.

De esto se encarga el método de cambio medio o “Mean Shift”, este método ubica el centroide de la detección y según la distribución de puntos o forma del objeto, asume una región de interés en la vecindad del objeto dentro de la imagen, ahora, en esta región de interés utiliza una función de semejanza, con la cual, compara los cuadros delimitadores en búsqueda de un cuadro delimitador con características similares al del fotograma anterior, de esta forma asigna una etiqueta al objeto y lo rastrea a lo largo del video. [45]
Este algoritmo es similar al algoritmo de *K-means*, respecto a que utiliza grupos o *clusters* basados en la distribución de los datos, para crear regiones de interés, una de sus desventajas es que, al utilizar una región según la forma de la imagen, cuando el objeto se mueve demasiado rápido y sale de esta región se pierde la etiqueta o ID, por ende, tenemos un error en el seguimiento del objeto.

3.8 Método de flujo óptico

Este método, también conocido como algoritmo de seguimiento “*Optical Flow*” de Lucas Kanade, este difiere bastante del anterior, ya que usamos el movimiento del objeto para realizar el seguimiento, no usamos las características del objeto, sino que este método crea unos vectores de movimiento del objeto a través de la imagen (véase Figura 17), utiliza las secuencias de fotogramas y según el brillo o intensidad de cada pixel en la imagen, genera un flujo de movimiento, con este flujo y asumiendo una velocidad constante es posible predecir la posición a futuro del objeto. [45]

Este algoritmo comúnmente se usa en conjunto con el método de *MeanShift*, de forma que se genere una región de interés más específica para rastrear el objeto en el fotograma actual, sufre del mismo problema relacionado con la velocidad del objeto, ya que a

velocidades altas lo pierde de vista, además de que es muy susceptible a la oclusión y a las trayectorias aleatorias. [46]

Figura 17. Vectores de desplazamiento con Optical Flow. [46]

3.9 Filtro de Kalman

Este método es de los más comunes en cuanto a seguimiento de objetos se refiere, aquí se utilizan las detecciones actuales y las anteriores, asumimos una velocidad constante y una distribución gaussiana, entonces cuando estamos realizando el seguimiento tomamos en cuenta el valor de confianza o puntaje del detector de que tenemos un objeto en el cuadro, de esta forma, cuando está seguro de que es un objeto simplemente usamos vectores de movimiento para determinar el ID del objeto, pero en caso de oclusión o ruido en la imagen, se reduce la confianza del algoritmo y toma mayor peso los vectores de movimiento del fotograma pasado, es equivalente a decir que nuestra región de interés se seguirá desplazando a la velocidad del objeto aunque el objeto no sea visible, de forma que cuando el objeto aparezca de nuevo en la imagen, se identifique correctamente y se le asigne la etiqueta correcta. [45]

3.10 DeepSORT

Eventualmente, para aplicación de detecciones en secuencias de imágenes (video) se diseñó DeepSORT, es una extensión de SORT (*Simple Online and Realtime Tracking*), un algoritmo que predice las trayectorias del objeto a lo largo del video, para de esta forma asignarle un ID o etiqueta especifica que permita identificarlo y diferenciarlo de los demás objetos estáticos o en movimiento dentro del video. Para ello es necesario primero realizar la detección del objeto, la cual en este caso está basada en YOLO v3, pues es el estado del arte para la detección a alta velocidad o en tiempo real, lo cual es bastante importante para llegar a procesar un video, pues este contiene varias imágenes en un solo segundo, entonces si queremos una procesar un video en tiempo real, necesitamos de la detección en tiempo real, con respecto a esta información predice la posible trayectoria o localización del objeto en el fotograma siguiente, con el objetivo de no perderlo de vista en ningún momento. [47] [48] [49]

Este método se apoya en el filtro de Kalman, aquí creamos un "*track*" o seguimiento para cada detección, este contiene la información de los centroide de cada cuadro delimitador, el área del cuadro, su relación de aspecto alto vs ancho y sus velocidades para cada componente (track={x,y,a,r,x',y',a',r'}), a medida pasan los fotogramas mantiene la información del objeto, cuando no observa el objeto en la detección, el *track* igualmente es actualizado según predicciones por modelos matemáticos asumiendo velocidad lineal constante hasta que supera el umbral de tiempo y elimina el *track* de dicho objeto.
Para la asociación de etiquetas a cada objeto se usa el algoritmo húngaro estándar, este utiliza una métrica de distancia para asociar los datos de la detección del fotograma actual y del seguimiento con DeepSORT, además de usar el IoU o área de intersección para comparar las detecciones actuales y determinar cuál se parece más según su posición y tamaño a la detección anterior. Aquellas detecciones que no se encuentren relacionadas con objetos del fotograma anterior se consideraran nuevos objetos en la imagen, por lo cual se crea un nuevo *track* con velocidad en cero y un ID correspondiente, aunque la covarianza para la región de interés será grande debido a esta incertidumbre de la velocidad del objeto.

Lo anterior aplica para SORT y DeepSORT, pero en este último agregamos una métrica según la apariencia del objeto conocido como descriptor de apariencia, el cual se logra con una CNN, y básicamente es un mapa de características del objeto con el cual se da una asociación más eficiente entre las detecciones y el *track* existente.

Este método es bastante mejor comparado con métodos como *tracktor*++, *track* R-CNN, JDE*, Mean Shift y Optical Flow*. Esto ya que posee mejor desempeño relacionado a eventos de oclusión en el video y a cambios de velocidad del objeto en la imagen, además de que es computacionalmente más simple, de forma que resulta ser mejor opción para aplicaciones en tiempo real. [47] [49]

Capítulo 4 - Metodología

En las secciones anteriores se ha llegado a dar un mayor entendimiento en lo relacionado a Inteligencia artificial y estado del arte respecto a métodos de detección y seguimiento de objetos en general, pero volviendo a nuestra pregunta de investigación basada en el contexto clínico y la problemática, ¿Cómo se realiza el seguimiento y detección de personas por medio del análisis de videos?, debemos considerar todas las opciones, dificultades, técnicas e hipótesis útiles para solucionar el problema planteado, el cual está relacionado con el hecho de que algunos pacientes con enfermedades neurodegenerativas o en proceso de rehabilitación, son observados manualmente por especialistas por medio de cámaras de vigilancia para llevar un control de sus actividades, posición y desplazamiento a lo largo del día, estamos hablando de que fácilmente pueden pasar horas observando un video para encontrar patrones que indiquen mejora o alerten sobre el paciente.

Para la elaboración del algoritmo de detección usamos YOLO v3 y DeepSORT, ya que son considerados métodos estándar para detección y seguimiento de objetos en tiempo real, con desempeño bastante superior a otros métodos, en términos de oclusión, precisión y principalmente comparándolos con su velocidad para procesar un video. Aunque actualmente para inicios del año 2021, ya se está empezando a usar YOLO v4 con DeepSORT gracias a su mejor desempeño, esta versión es bastante reciente y existe más información de su versión anterior YOLO v3.

Para el algoritmo, se utiliza una versión adaptada a *Google Colab*, del autor Pylessons, la cual contiene el algoritmo de YOLO v3 con DeepSORT, además, contiene una versión prematura de YOLO v4, de la cual no ofrece su máximo rendimiento, también tiene versiones *Tiny*, las cuales son menos potentes, pero ofrecen mejor rendimiento en tiempo. Este algoritmo se encuentra disponible en GitHub, https://github.com/pythonlessons/TensorFlow-2.x-YOLOv3, básicamente se trabaja sobre este programa, de forma que se adecúe al seguimiento exclusivo de personas, y finalmente permita observar factores de desplazamiento como la distancia recorrida, la trayectoria y velocidad en el video. Un punto a favor de utilizar máquinas virtuales de Colab, es que se pueden conectar a plataformas como *Google Drive* y *Google sheets*, de los cuales se hablará más adelante.

En el presente proyecto se plantea generar un algoritmo o código basado en modelos de inteligencia artificial que permitan la identificación de personas y el monitoreo en tiempo real para ayudar a detectar posición y cuantificación del gasto calórico según se requiera, adecuado de algoritmos para al reconocimiento de objetos o seguimiento de multitudes. Para esto, se plantean varias actividades a realizar:

4.1 Base de datos de videos de personas

La data de trabajo se divide principalmente en 2, un grupo de videos cortos descargados directamente desde *YouTube* donde se observan personas caminando en diferentes escenarios, este grupo consta de 3 videos de no más de 30 segundos con los cuales se realizó pruebas y configuraciones en un principio. Ya el segundo grupo, son 10 videos de la base de datos “EPFL data set”, de donde obtenemos 5 videos en interiores (oficina y

estacionamiento) y otros 5 videos en exteriores (Campus universitario y terraza). Sobre la base de datos EPFL podemos decir que son videos tomados desde diferentes cámaras para el mismo evento, en este caso peatones caminando, donde los peatones son miembros del laboratorio, cada cámara está ubicada 2 metros sobre el suelo y el objetivo de estos videos fue analizar su propio algoritmo de detección y seguimiento de personas: *Multi-Camera People Tracking with a Probabilistic Occupancy Map.* [50]

Respecto a la selección de los videos, cada grupo de videos fue seleccionado respecto a distintas etapas evolutivas que presentó el algoritmo, empezando desde los videos de prueba, que son aquellos con los cuales se testeó el algoritmo para facilitar el análisis de cada etapa, componentes y comportamiento en general de la red neuronal, hasta los videos de la base de datos EPFL, con los cuales se corrobora el análisis de multitudes y se observa cómo influyen los cambios de ángulo en la cámara. En la fase de prueba se utilizaron 3 videos, los cuales se describen a continuación:

- **Personas caminando en Londres** [51]

En este caso el video original tiene una duración de 37 minutos, del cual se eligió un segmento de 29 segundos en donde se observan varias personas y diferentes objetos en la zona de Piccadilly Circus en Londres en un ambiente lluvioso y de noche. Este video se encuentra a resolución HD de 1280x720 p, con una tasa de 30 FPS. Lo que se busca es analizar el comportamiento del algoritmo frente a cambios dinámicos de la posición de la cámara en un ambiente oscuro, así como dar a entender la diferencia entre detección y seguimiento en un mismo video.

- **Mujer huyendo de su vehículo** [52]

En este video vemos una mujer adulta corriendo, ya que al parecer una paloma se había colado en su vehículo, respecto al video está a una tasa de 30 FPS, con resolución HD de 1280x720 p, aquí solamente contamos con 2 segundos de video, suficiente para hacer el seguimiento y graficar la trayectoria de una sola persona. Con este video se comenzó las pruebas para obtener el mapa de calor y la trayectoria de cada sujeto, pues al tener una sola persona facilita su entendimiento.

- **Sospechosos: Departamento de Policía de Lexington**. [53]

En este video se observa las imágenes captadas por una cámara de seguridad doméstica, en donde se ve a dos jóvenes acercándose a la puerta principal. Después de notar la cámara, las personas inmediatamente se dieron la vuelta y se fueron. Este video surge luego de que varios residentes del vecindario hayan informado de robos o intentos de allanamiento. En este caso usamos un corte de 9 segundos, igualmente a resolución HD 1280x720 p, con tasa de 30 FPS. Con este video, se planea dar el siguiente paso para la elaboración del algoritmo, de forma que podamos obtener un valor de distancia recorrida para cada persona, que, a su vez de manera proporcional, indica el gasto calórico y según el tiempo transcurrido se llega a tener una percepción de la velocidad de los individuos en el video.

En la Figura 18 se observa un ejemplo de las imágenes que contiene cada uno de los videos.

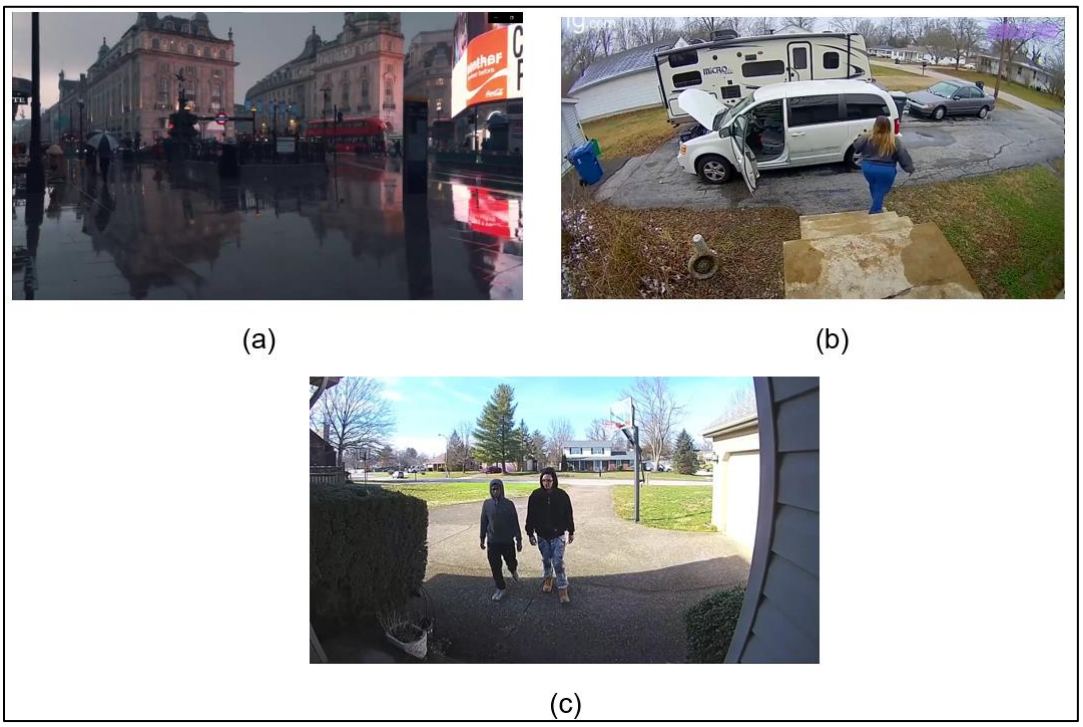

Figura 18. Videos de prueba.
*. **(a)** Londres [51], **(b)** Mujer huye del vehiculo [52], **(c)** Sospechosos Lexington [53]*

4.2 Base de datos EPFL [50]

Finalmente, con los videos de la **base de datos EPFL** *(École polytechnique fédérale de Lausanne* ó la Escuela Politécnica Federal de Lausana*),* se puede analizar el desempeño del algoritmo en velocidad y precisión, pues son grupos de videos en situaciones controladas, que fueron intencionalmente diseñados para dificultar la detección y seguimiento de personas con algoritmos de este estilo, con gran número de oclusiones y a una resolución de apenas 360x288 pixeles, es decir, contiene menor información para la detección del objeto. En este caso, la selección tiene 2 categorías, videos en ambientes cerrados, específicamente 3 videos de un laboratorio de la Escuela Politécnica Federal de Lausana y 2 videos de un pasadizo debajo de una estación de tren en suiza; ya en la categoría de videos en ambientes abiertos tenemos 3 videos tomados en el campus de la misma institución educativa, y 2 videos tomados desde la terraza de la misma universidad. Cada subgrupo posee la misma secuencia de los sujetos, en donde caminan de forma aleatoria pero vistos desde diferentes cámaras, nombrados como se encuentra a continuación:

Videos en ambiente cerrado o *Indoor*

- Grupo 1c, 4p-c0 → 4 personas cámara 0 en el laboratorio
- Grupo 1c, 4p-c1 → 4 personas cámara 1 en el laboratorio
- Grupo 1c, 6p-c1 → 6 personas cámara 1 en el laboratorio

- Grupo 2c, pasillo1-c1 → varias personas en el pasadizo de la estación de tren, cámara 1
- Grupo 2c, pasillo1-c2 → varias personas en el pasadizo de la estación de tren, cámara 2

Videos en ambiente abierto o *outdoor*

- Grupo 1a, campus4-c0 → más de 4 personas en el campus, cámara 0
- Grupo 1a, campus4-c1 → más de 4 personas en el campus, cámara 1
- Grupo 1a, campus4-c2 → más de 4 personas en el campus, cámara 2
- Grupo 2a, terrace7-c1→ más de 7 personas en la terraza, cámara 1
- Grupo 2a, terrace7-c3→ más de 7 personas en la terraza, cámara 3

A continuación, se observan unas imágenes de ilustración para comprender cada uno de los videos.

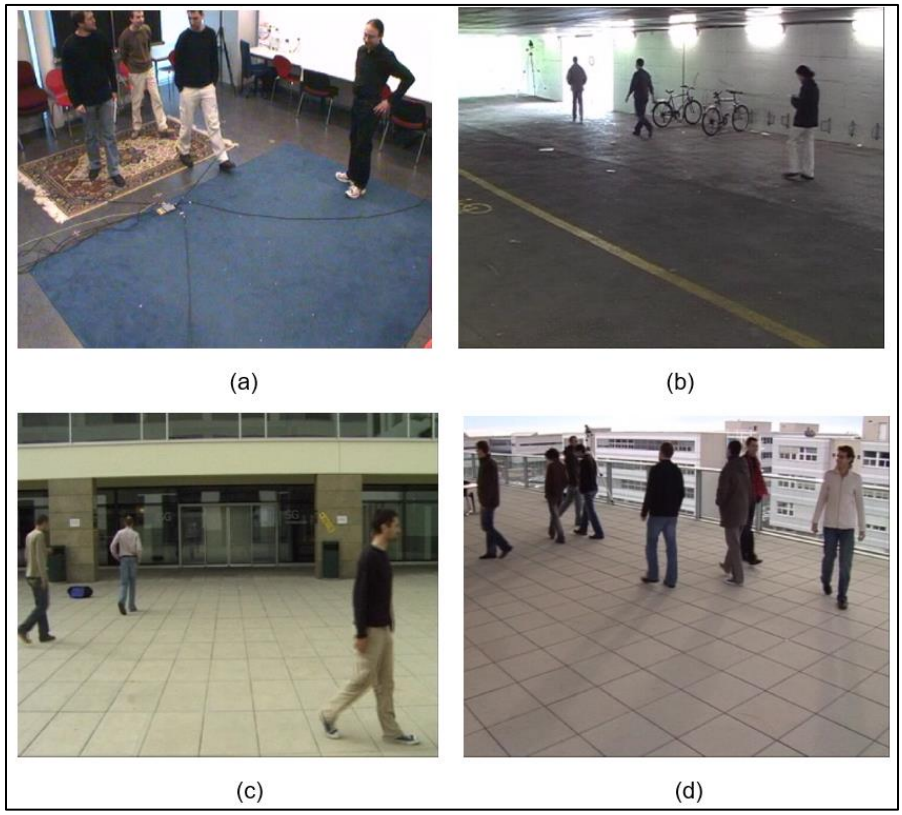

Figura 19. Videos base de datos EPFL. [50]
(a)*Grupo 1c,* ***(b)****Grupo 2c,* ***(c)****Grupo 1a,* ***(d)****Grupo 2a*

4.3 Preprocesamiento del video

Para la detección es fundamental contar con material de videovigilancia, el cual debe encontrarse en formato MP4, este es un formato estándar muy utilizado para almacenar videos, el programa no solamente trabaja con videos descargados, sino que también permite emplear la cámara web propia del computador, que en nuestro enfoque

corresponde a la cámara del centro de salud en tiempo real. En esta etapa de preprocesamiento se tiene en cuenta la fase de calibración, en donde se debe transformar la perspectiva del video, con el fin de obtener un plano o vista de las dimensiones que facilite el cálculo de las distancias (recordemos que los videos son monoculares, es decir, tomados de una sola cámara), para ello, consideramos como referencia algún objeto dentro de la imagen, para el video de prueba de los sospechosos del departamento de Lexington se utiliza la cesta de baloncesto como referencia asumiendo una altura de 3,05m (medida estándar utilizada en NBA) [54], entonces desde el algoritmo encontramos la distancia marcada en rojo en la Figura 20, esta distancia se encuentra en pixeles, ya que se utilizan las coordenadas de los puntos x1,y1 y x2,y2 correspondientes al punto sobre el suelo y el punto a nivel del aro. Ahora usamos las siguientes ecuaciones para hallar la distancia entre los puntos y para convertir esa distancia de coordenadas a distancia en metros:

$$Longitud\ de\ referencia\ en\ pixeles\ = \sqrt[2]{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad (Ec. 1)$$

$$Distancia\ en\ metros = \frac{(Longitud\ medida\ en\ pixeles) \cdot (3{,}05\ metros)}{Longitud\ de\ referencia\ en\ pixeles} \qquad (Ec. 2)$$

Figura 20. Calibración de longitudes en Video de prueba.

Esta es una aproximación bastante empírica, pues no existe un método de oro para obtener las dimensiones de objetos en un video, normalmente en aplicaciones de este estilo, se cuenta con puntos de control de todo el encuadre del video, luego se utiliza la función *getPerspectiveTransform*() de *openCV*, que transforma la perspectiva del video a una vista de pájaro basándose en 4 puntos en el suelo, que sean conocidos en el video. Existen métodos como el "modelo *pinhole" o también el modelo Tsi,* que de igual forma necesitan de una calibración previa de la cámara y unos puntos de control o referencia para obtener un plano de coordenadas apropiado, en nuestro caso, para el enfoque del proyecto, no se busca calibrar tantos parámetros, sino que es una aproximación bastante simple, para que cualquiera pueda utilizar el algoritmo. [55]

En la base de datos EPFL se busca también reducir su complejidad, ya que no contamos con las dimensiones del laboratorio, usamos la altura de las personas como valor de referencia, se usa el promedio de altura por regiones, en este caso la altura promedio de un hombre colombiano es de 1.70 metros, este valor es utilizado en los videos posteriores para establecer una métrica más generalizada, de forma que aumenta de cierta forma la autonomía del algoritmo. [56]

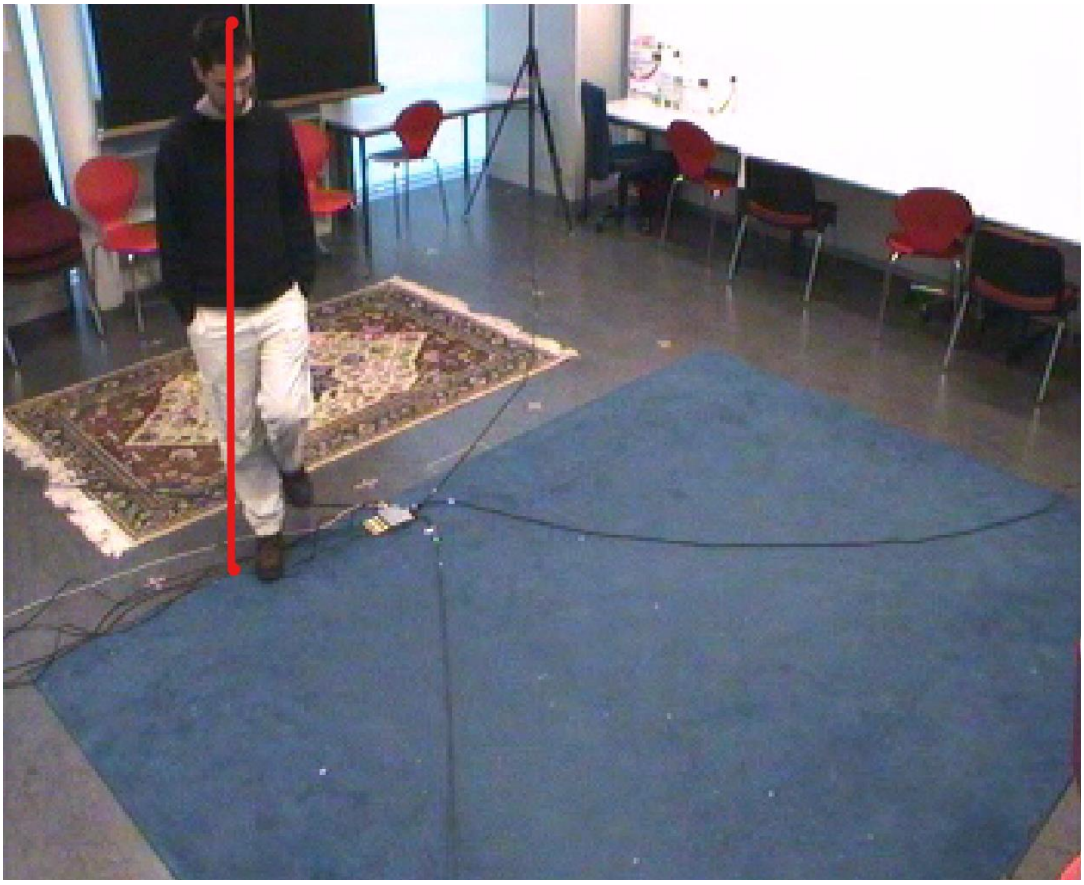

Figura 21. Calibración de videos EPFL.

Por ende, con esta calibración de los videos se puede obtener un valor en metros para la trayectoria de cada persona, que, en teoría, debería ser la misma en un grupo de videos, pues estamos hablando de que las personas están realizando la misma trayectoria, pero lo observamos desde distintas cámaras.

4.4 Detección, Seguimiento y monitoreo de personas en tiempo real

Para desarrollar el algoritmo usaremos principalmente el lenguaje de programación de Python, apoyado en algoritmos de *Deep learning* disponibles en internet, entre ellos uno de los más atractivos para la detección de objetos es el programa YOLO v3, adicional es necesario un programa que realice el seguimiento del objetivo, en este caso se utiliza el programa DeepSORT, ya que es un programa muy popular y de los más utilizados debido a su gran desempeño ante problemas de oclusión. En la detección de objetos se analiza

cada fotograma de forma independiente y se procesa en búsqueda de las personas de dicho fotograma en particular, y para el seguimiento de objetos, necesita rastrearse dicha persona en la serie de fotogramas (video), esto se hace basándonos en la información del fotograma pasado y asignando una identificación ID a cada persona para luego buscarla en la información actual, de esta forma se identifica la posición del sujeto en la imagen o video, complementando con el filtro de Kalman, que nos ayuda a tener en cuenta el ruido en la detección y utiliza el estado anterior para predecir un buen ajuste para los cuadros delimitadores de cada objeto, ya que a partir de esta delimitación del objeto o persona en nuestro caso, se realizan los análisis de la trayectoria de movimiento. [45]

Cuando hablamos de monitoreo, hace referencia al personal médico encargado del paciente, ya que, en este enfoque, este algoritmo es una herramienta para facilitarle su labor. Como tal, el algoritmo no tiene la capacidad de generar alertas, diagnósticos o alarmas, simplemente arroja información que puede ser de gran ayuda a los profesionales en rehabilitación o terapia física. Pues como se menciona a continuación, es necesario trabajar en máquinas locales, o en un ordenador propio para obtener un mejor desempeño, de forma que se pueda obtener información en tiempo real, con la cual generar algunas alertas según sea necesario.

Respecto al procesamiento de video en tiempo real, se realiza un análisis observando el tiempo de procesamiento del video, ya que en general, el simple hecho de realizar *streaming*, es decir, usar la información del video en tiempo real mientras se carga y descarga cada fotograma del video desde la red a través de máquinas virtuales de *Google Colab* nos genera una latencia considerable. Esto gracias a las cantidades de información y el procesamiento que se lleva a cabo en el algoritmo, lo cual hace prácticamente imposible el realizar aplicaciones en tiempo real vía internet, por lo menos en una región como Colombia, en donde se dispone de velocidades 4G y de las peores infraestructuras digitales de Latinoamérica [57]. Por lo tanto, se dispone en el algoritmo, de una sección que contabiliza el tiempo de procesamiento interno de la máquina virtual de *Google Colab*, esto apoyándose en la función *.time()* de Python, de esta forma se llega a tener una idea de los Fotogramas que se procesan en cada segundo, una tasa imprescindible si de desea conocer el desempeño del algoritmo para aplicaciones en tiempo real. Este valor es mostrado directamente sobre el video de salida, como valor de FPS y FPS total, la diferencia es que el primer valor corresponde solamente al tiempo de ejecución de la detección con la red neuronal de YOLOV3, en cambio el valor de FPS total incluye el seguimiento con DeepSORT, es decir el tiempo total desde que ingresa la imagen a la red neuronal y sale con su ID, cuadros delimitadores, velocidad, distancia y calorías para cada persona.

- **Creación de gráficos y mapa de calor.**

Fue en la etapa de videos de prueba, específicamente con el video *mujer huyendo se su vehículo*, aquí la mujer se desplaza de izquierda a derecha como indica la flecha en la Figura 22, en esta etapa se encuentran los centroides de cada persona, entiéndase por centroide como el punto central del cuadro delimitador para cada persona detectada, luego se comienzan a graficar sobre un fondo blanco, de aquí tenemos un gráfico que nos muestra una dispersión de puntos de los sujetos a lo largo del video (véase Figura 23), esta información de las coordenadas (x,y) de los centroides posteriormente pasa a la creación del mapa de calor, en donde la data adquiere mayor significado, pues en el mapa de calor

se tiene un mapa de frecuencia, en donde las secciones rojas son aquellas en donde hubo mayor concurrencia de personas, y las zonas azules aquellas donde hubo presencia baja o nula de personas en el video.

Figura 22. Movimiento de la mujer en el video. (Desde el cuadro 1 hacia el cuadro 4)

Para la adquisición del mapa de calor, se utiliza la función *histogram2d* y un filtro gaussiano con sigma = 16 gracias a la función *gaussian_filter* de scipy, este conjunto de funciones nos arroja una figura de frecuencia, que indica el recorrido del sujeto y los puntos más concurridos por los transeúntes. A medida que aumenta el valor de sigma, el filtro realiza un mayor suavizado, por lo cual tiende a verse más borrosa la imagen, este filtro es modifica directamente los pixeles de la imagen del Histograma, de forma que atenúa los bordes . Además, si se observa bien la Figura 23, en el gráfico de trayectoria de la izquierda, tenemos dos puntos en esquinas opuestas, arriba a la izquierda y abajo a la derecha, estos puntos permiten observar la imagen completa a su resolución original, pues la función de histograma2D no asigna colores suficientes para abarcar toda la imagen, solo regiones con valores no nulos y su cercanía, es decir, veríamos solamente una región del mapa de calor (la región “caliente”) y no la imagen completa.

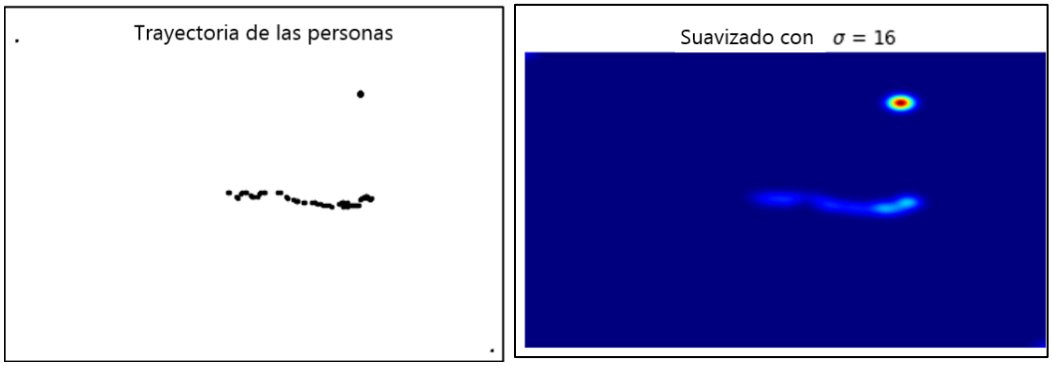

Figura 23. [Izquierda] Trayectoria recorrida por una persona en el video. [Derecha] Mapa de calor de trayectorias del video.

Adicionalmente, es en esta sección en donde se identifica con colores las trayectorias que realizan las personas en el video, ya que con el mapa de calor tenemos una figura representativa de la posición de todos los sujetos en el video en general, en cambio con las trayectorias individuales se puede entender el comportamiento especifico de cada individuo, en este caso se va graficando para cada ID de persona un color distinto, como se observa en la Figura 24.

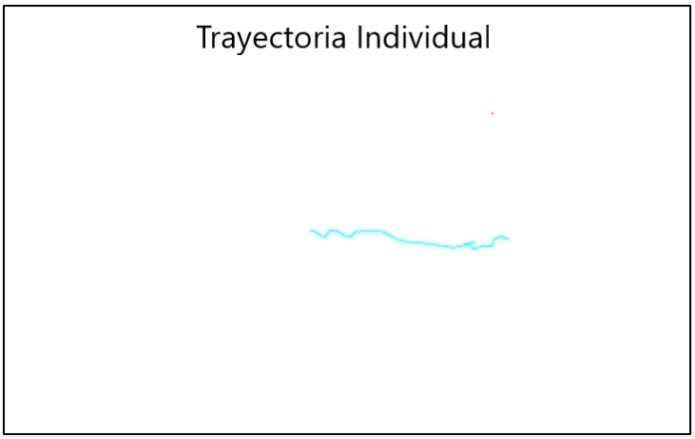

Figura 24. Trayectoria a color de la persona.

- **Cálculo de velocidad, distancia y calorías**

Para esta etapa, se sigue en la evolución del programa, el video utilizado fue *sujetos sospechosos de Lexington*, aquí avanzamos hacia calcular la distancia entre los centroides de la imagen anterior y la imagen actual para la misma persona, esta distancia en pixeles se convierte a distancia en metros con una regla de 3 simple y con ayuda del objeto de referencia como la altura del aro de baloncesto, entonces si *n* pixeles son 3 metros, a cuando metros corresponde mi distancia en pixeles (Ecuación 2), una vez tenemos la distancia recorrida por la persona en metros, usamos la ecuación 4 para obtener el gasto calórico de cada persona a lo largo del video, convirtiendo la distancia recorrida (en metros) en calorías, para la ecuación no contamos con un valor del peso de cada sujeto en los videos, por lo tanto, supondremos una media de 67,9 Kg para Latinoamérica, según *BMC Public Health*. [58]

$$\text{Kca} = (0.7768) \cdot (\text{distancia}[\text{Km}]) \cdot (\text{Peso}[Kg]) \quad (Ec.3)\ [59]$$
ó
$$\text{ca} = (0.7768) \cdot (\text{distancia}[m]) \cdot (67{,}9\ \text{Kg}) \quad (Ec.4)$$

Como última instancia, para tener un valor de la velocidad de cada persona en el video, usamos la tasa de FPS de cada video, se tiene la distancia recorrida por cada persona sobre el tiempo transcurrido en 0.5 segundos, ya que dependiendo de la tasa de fotogramas del video, que por lo general es de 30 FPS, se conoce el tiempo que hay entre un fotograma y el siguiente, entonces en este caso donde tenemos 30 imágenes en 1 segundo (30 FPS), si queremos calcular la velocidad cada 0.5 segundos, debemos tomar la distancia recorrida y el tiempo que hay entre la imagen de inicio y el fotograma No. 15, lo que corresponde a medio segundo (Velocidad= Distancia/tiempo), este valor de velocidad se dibuja directamente en la imagen actual al lado del ID que posee la persona, de forma que, en el video de salida, tenemos a lo largo del tiempo un valor de velocidad instantánea, que recordemos, depende de las distancias calculadas previamente para cada persona.

4.5 Generación de alertas y reportes consecuentes según los datos obtenidos

Con la información obtenida gracias al algoritmo se busca dar una herramienta que facilite la comprensión de videos, en un principio con un enfoque médico llevado a la rehabilitación. Los profesionales de la salud o fisioterapeutas se pueden apoyar de este tipo de tecnologías para determinar si el paciente se ejercitó lo suficiente, así como de su condición física en general. Ya que gracias a estos programas, se puede observar en el gráfico de trayectoria el recorrido realizado por el paciente, con el fin de sugerir cambios en la conducta de los pacientes, alertar de posibles cambios de velocidad, posición o cuando el paciente abandona el lugar, pues el propósito de este programa no es dar algún diagnóstico médico, simplemente ayudar al personal para gestionar su tiempo en otras tareas y facilitar su labor, mientras tanto, el algoritmo arroja unos resultados que pueden dar una descripción general de lo ocurrido.

4.6 Validación de la confiabilidad del sistema desarrollada mediante usando bases de datos de personas EPFL

Con el set de datos de EPFL se organiza la información de los videos, vemos cómo afecta la oclusión al algoritmo y a su vez, la influencia de procesar videos en ambientes cerrados como el laboratorio y en ambientes abierto como el campus, en esta etapa el algoritmo posee todas las funciones mencionadas con anterioridad. Para ello, en cada grupo de videos (1c, 2c, 1a, 2a) se tiene una calibración diferente, aunque en todas se utiliza la altura de referencia de 1.70 metros de un colombiano promedio, se eligen cuadros distintos dentro de cada video, por ende tenemos diferentes relaciones entre la trayectoria de la imagen en pixeles y la trayectoria real en metros, otro cambio que se realizó para tener una mejor representación de los datos de distancia y calorías para cada persona, fue la implementación de *Google sheets*, gracias a la librería *gspread* tenemos acceso a las hojas de nuestro perfil de *Google sheets*, para ir agregando los resultados en un Excel, este registro de los datos aumenta el tiempo de procesamiento general del algoritmo, pues dicha librería solo permite escritura de hasta 500 datos por minuto, escribir más datos conlleva a un error en la ejecución del programa debido a un agotamiento, por ende se utiliza un *delay* con la función *time.sleep(150)* que permita descansar al servidor durante 150 segundos.

Capítulo 5 - Resultados

Una vez implementado el algoritmo de detección y seguimiento de personas, se procede a realizar un análisis de desempeño para diferentes situaciones, entre las cuales se destacan cambios en el número de personas en el video, ubicación y tiempo. Como primera instancia se comienza realizando pruebas con videos cortos, de no más de 30 segundos, con los cuales se comprueba el correcto funcionamiento del algoritmo YOLOV3 para detección de sujetos.

La idea principal de utilizar estos videos es evidenciar el funcionamiento de YOLO v3 con DeepSORT en diferentes escenarios e ir configurando las funciones adicionales para cada etapa del procesamiento. A continuación, una breve descripción de cada video y de lo que se evidencia al analizarlos con nuestro algoritmo.

5.1 Personas caminando en Londres

Detectando todas las clases en el video el tiempo de procesamiento fue 1m 49s para la detección desde *Google Colab*, aquí el algoritmo de YOLO v3 detectó clases como, por ejemplo, las personas, las sombrillas, los autos, buses, bicicletas y semáforos. (Véase Figura 25), además, podemos observar el puntaje o probabilidad asignado a cada objeto en la imagen, junto con los FPS en la parte superior izquierda, este valor nos indica que para este video estuvo procesando **12,7 FPS**. Este resultado se obtiene usando la función *time* de Python, que nos calcula el tiempo transcurrido dentro del sistema desde que se llamó el modelo YOLO y nos arrojó las predicciones.

Pero lo que observamos desde *Colab* sería que detectó todos los fotogramas del video (29seg x 30 FPS= 870 Fotogramas) en un tiempo de procesamiento de 1m 49 seg, lo cual nos da una tasa de **7,98 FPS**. Es decir, el primer valor resaltado corresponde al calculado por la máquina virtual, pero el segundo es el que toma en realidad para nosotros como observadores.

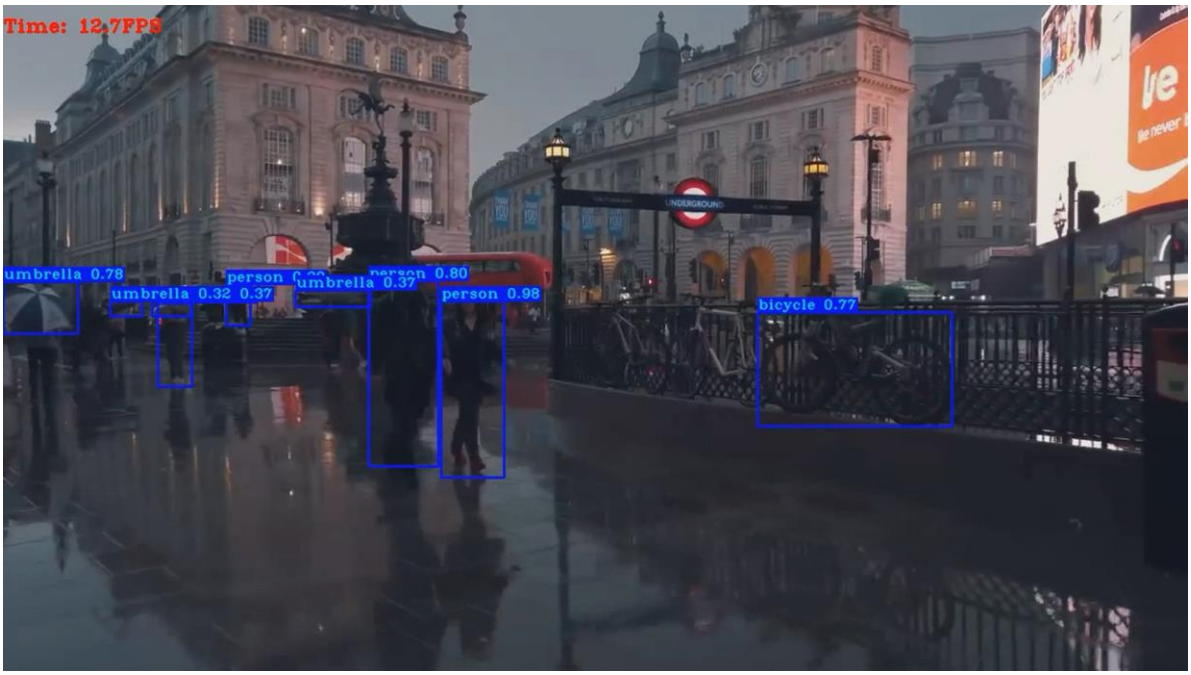

Figura 25. Detección con YOLO V3.

Otro aspecto que destaca es que, aunque es de noche y está lloviendo, la detección se hace bastante bien, a una tasa aceptable de FPS y con gran cantidad de objetos, ahora debemos revisar el seguimiento, en nuestro caso solamente de personas. Para ello ejecutamos el algoritmo de seguimiento basado en DeepSORT, (Véase Figura 26), en este caso el resultado se demora un tiempo de 1m 58s para el procesamiento total del video, en la imagen ya no tenemos la probabilidad del detector, sino que tenemos la clase "*person*" junto con un ID asignado a dicha persona, que en teoría debería mantenerse para la misma persona a lo largo del video.

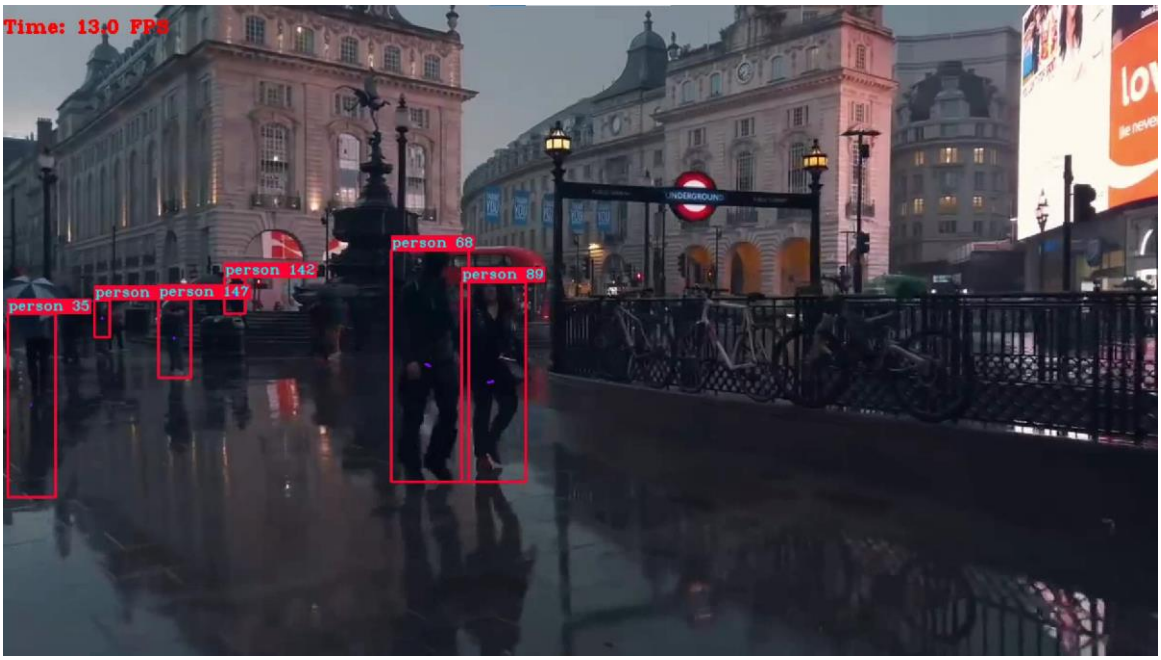

Figura 26. Seguimiento con DeepSORT.

La tasa de FPS que se muestra en la imagen, de 13 FPS, es la que posee el detector de YOLO v3 para este video, pero con respecto a la tasa total, incluyendo al algoritmo de seguimiento DeepSORT y los demás cálculos adicionados, tenemos un valor promedio de **9,1 FPS en total**. Aunque si tomamos el total de fotogramas del video (29seg x 30 FPS= 870 Fotogramas), y lo dividimos en el tiempo que tomó procesar el video en *Colab* (1m 58 seg) tenemos una tasa de **7,37 FPS en total**.

Lo anterior se encuentra condensado en la Tabla 1:

	FPS
YOLO v3	12
YOLO v3 desde Google Colab	7,98
YOLOv3 y DeepSORT	9,1
YOLOv3 y DeepSORT desde Google Colab	7,37

Tabla 1. Valores en FPS obtenidos para detección y seguimiento

5.2 Señora huyendo de su vehículo

En este caso, debido a que es solo una persona (Véase Figura 27), la velocidad es de 13,1 FPS para la detección, pero con el seguimiento fue de 10,6 FPS en total, a la vista solamente tenemos una persona ("*person* 1"), pero el algoritmo está detectando al fondo a otra persona ("*person* 2"), que en realidad es un buzón, esto es un falso positivo y se observa también más adelante en los gráficos de trayectoria y mapa de calor.

Figura 27. Detección y seguimiento de un solo sujeto.

5.3 Sospechosos: Departamento de Policía de Lexington.

Con este video buscamos escalar ahora a una imagen con más personas respecto al video anterior, para ver el desempeño del algoritmo paso a paso, además, aquí ya nos enfrentamos al primer problema, pues en medio del video ocurre una oclusión significante.

Para llegar a tener una idea de la distancia recorrida, se dibuja una línea en pixeles, relacionada con una distancia en metros, de forma que en etapas posteriores se puede tener una trayectoria física de la persona. Cuando establecemos la calibración con la altura del aro, para establecer a cuantos pixeles corresponde un metro, nos da como resultado 78,33 pixeles cada metro de longitud. Con este valor se calcularon los valores de distancia y calorías de cada sujeto que se encuentran en la Tabla 2.

ID	Distancia (m)	Calorías
persona 1	12,20	643,65
persona 2	6,02	317,60
persona 3	0,21	11,15

Tabla 2. Valores obtenidos del Video de prueba.

Además, en esta etapa se obtuvo la velocidad para cada persona en el video, como se observa en la Figura 28, donde en cada cuadro delimitador tenemos la clase "*person*" con su ID correspondiente y justo al lado, el valor de velocidad **V** en m/s para ese instante en el video, esta información es útil para por ejemplo determinar cuando ocurre una caída o el inicio de actividad física intensa, pues en caso de superar un umbral de velocidad se podría llegar a considerar como anormal.

Figura 28. Velocidades de los sujetos en el video.

5.4 Base de datos EPFL

Similar a la Tabla 2, para cada uno de los 10 videos seleccionados de la base de datos EPFL, se calcularon los valores de distancia y calorías que recorren los sujetos, se obtuvieron los mapas de calor y trayectorias para cada uno de los videos. Como ejemplo tenemos el video del pasadizo o pasillo bajo la estación de tren en suiza, de la cual, similar a los demás videos de la base de datos EPFL, se realizó la calibración en base a la altura de un transeúnte en el video (Figura 29, a), posterior se obtiene el mapa de calor (Figura 29, b) y la trayectoria por colores de los sujetos (Figura 29, c).

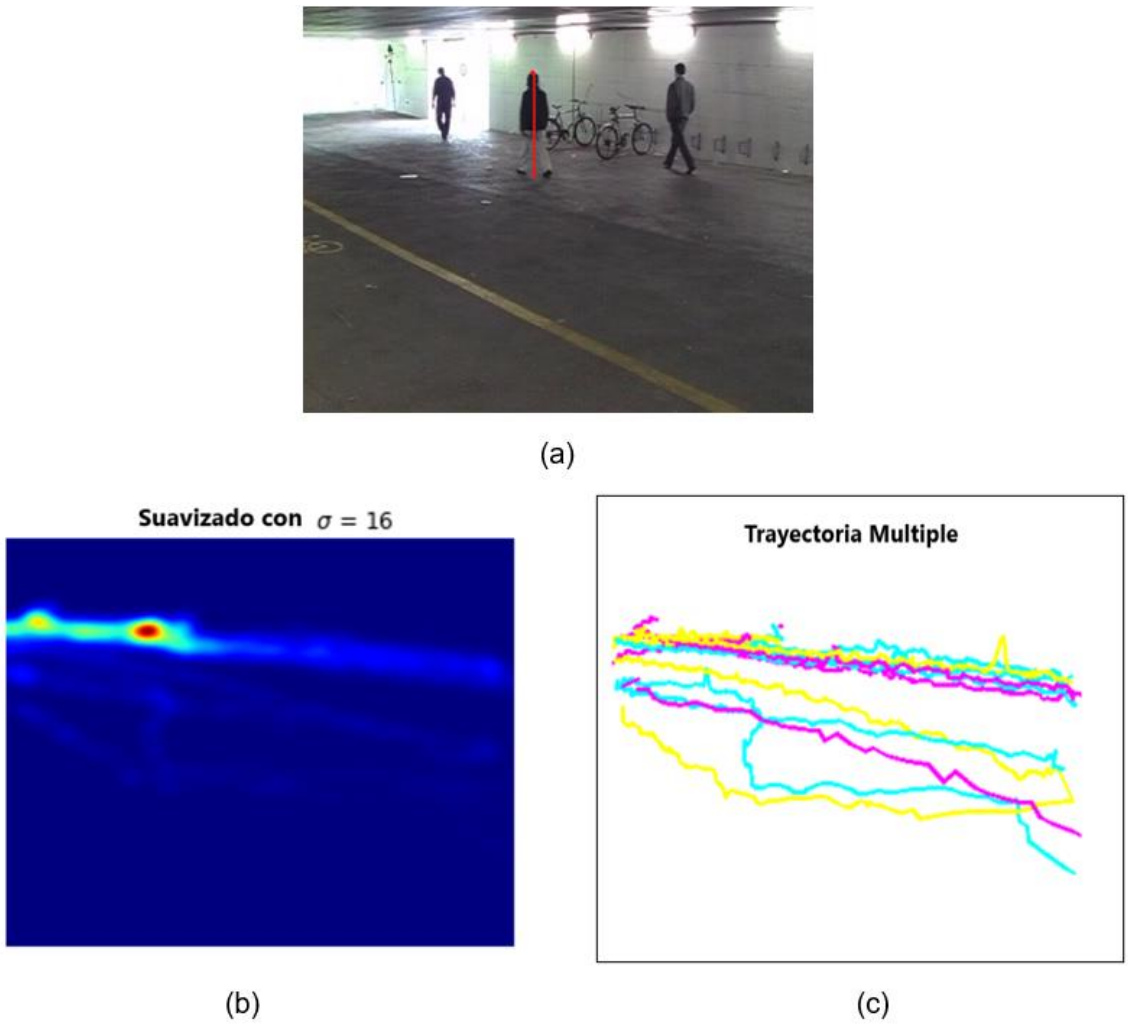

Figura 29. Resultados gráficos de video EPFL.
(a) Calibración (b) Mapa de calor (c) Trayectoria de las personas

En este caso, la información recibida del mapa de calor nos indica un punto rojo bastante concurrido a lo largo del video, lo cual es correcto, pues este punto corresponde a la salida del pasadizo, por donde la mayoría de las personas en el video entran y salen de la estación de tren.

- **Análisis de Resultados Clínicos**

Respecto a los datos aprovechables por fisioterapeutas, tenemos distintas variables, las cuales fueron igualmente calculadas para todos los videos de la base de datos EPFL y los videos de prueba, el objetivo de estos resultados es de dar información que facilite el monitoreo y diagnóstico por parte de los profesionales de la salud. Para este caso tenemos:

- **Mapa de calor** (Véase Figura 29)

Este grafico nos muestra el desplazamiento de los sujetos en el video, nos enseña los puntos donde fue más frecuente encontrar personas, en el caso de fisioterapia, sirve para

tener de cierta forma un resumen de lo sucedido a lo largo del video, pues recordemos que a medida el punto se torna de color rojo es probable que el paciente se haya detenido, así conociendo el punto y el momento se puede llegar a identificar el evento que cambio de cierta forma la rutina de ejercicio y detuvo el entrenamiento. Es decir que con este grafico podemos identificar fácilmente la inmovilidad de un paciente a lo largo del video.

- **Trayectoria** (Véase Figura 29)

Este grafico es ideal para conocer la posición del paciente, ya que nos muestra el recorrido realizado por el mismo en la secuencia de video, si por ejemplo estuviéramos analizando un solo paciente en el video, y observamos un cambio en el color de la trayectoria, nos indica que el paciente abandono y volvió al campo de visión de la cámara o que ingreso otra persona. Pues esta trayectoria nos muestra el punto de inicio y el punto final dentro del encuadre

- **Distancia Recorrida** (Véase Tabla 2)

En cuanto a la distancia recorrida, este puede ser un mecanismo de seguimiento para el cumplimiento de metas, por ejemplo, es más probable que el fisioterapeuta asigne como actividad recorrer el campo de futbol 10 veces o 3 Km, una forma de saber que se cumplió el objetivo es observando la distancia, pues en ese caso tendríamos como resultado que el paciente se desplazco 10 veces la longitud del campo de futbol.

- **Gasto calórico** (Véase Tabla 2)

Este valor se muestra en forma de tabla al finalizar el procesamiento del video, en un contexto para aplicaciones en tiempo real sería necesario mostrar el valor actual para llegar a un objetivo en el entrenamiento.

Con la cuantificación de las calorías tenemos una idea del ejercicio físico realizado por el paciente, ya que es un indicador directo de la exigencia del ejercicio, con este valor se puede determinar o no un cambio en las rutinas de ejercicio, para aumentar la dificultad y apoyar el proceso de rehabilitación en pacientes con alguna discapacidad.

- **Velocidad de los sujetos en el video** (Véase Figura 28)

La velocidad se muestra como un valor en metros por segundo encima de la etiqueta del paciente, que constantemente se va actualizando, a una tasa de 0,5 seg. La velocidad igualmente nos indica el nivel de desempeño por parte del paciente ya que se relaciona con la intensidad del entrenamiento, puede ser indicativo del modo de entrenamiento, pues nos indica si el paciente se encuentra caminando o trotando, así como también si lo está haciendo de forma intercalada.

Capítulo 6 - Discusión de resultados

El análisis de la precisión de los algoritmos implementados requiere de la comparación basados en porcentajes de error, en cuyo caso tenemos dos errores diferentes registrados en la Tabla 3 y Tabla 4.

El primero es el *porcentaje de falsos positivos*, que es el porcentaje de personas que fueron detectadas, pero que corresponden a un ID anterior, ya que uno de los errores más presentes en la data de salida es que, por ejemplo, en el video *Indoor* 4p-c0 solamente tenemos 4 personas en el video, que, si bien entran y salen algunas veces del encuadre de la cámara, no corresponde con la cantidad de ID de las personas identificadas, es decir, que en este caso donde tenemos 4 "personas reales" el algoritmo asigna 130 ID, es decir, según el algoritmo, en el video tuvimos 130 personas diferentes, de aquí que tengamos un error del 96,92%, ya que de esas 130 personas identificadas, 126 en realidad son resultado del fallo de asignación de etiquetas del algoritmo.

Como segundo valor importante, tenemos el *porcentaje de ceros o apariciones espontaneas*, este valor indica el porcentaje de personas identificadas, de las cuales no se realiza seguimiento o se pierde de vista, aquí nuevamente es un error en la asignación de etiquetas, ya que siguiendo con el mismo ejemplo del video *Indoor* 4p-c0, de las 130 personas identificadas por el algoritmo, a 76 de ellas, no se les volvió a encontrar en el video, por ende no poseen una distancia recorrida, lo cual se ve representado como un cero en el valor de calorías consumidas. En este caso el porcentaje de ceros es de 58,46% ya que un poco más de la mitad de las personas identificadas en el video 4p-c0 no poseen valor de distancia recorrida, por ende, no poseen valor de calorías consumidas.

Indoor	Personas identificadas	Personas reales	Personas sin seguimiento	Porcentaje falsos positivos	Porcentaje de ceros	Duración video
4p-c0	130	4	76	96,92%	58,46%	156 seg
4p-c1	211	4	147	98,10%	69,66%	156 seg
6p-c1	195	6	136	96,92%	69,74%	118 seg
pasillo1-c1	21	+7	4	66,66%	19,04%	100 seg
pasillo1-c2	18	+7	3	61,11%	16,66%	100 seg
Promedio	115	5,6	73,2	83,94%	46,71%	126 seg

Tabla 3. Valores obtenidos para los videos en ambiente cerrado.

Outdoor	Personas identificadas	Personas reales	Personas sin seguimiento	Porcentaje falsos positivos	Porcentaje de ceros	Duración video
campus4-c0	18	+4	5	77,77%	27,77%	79 seg
campus4-c1	26	+4	12	84,61%	46,15%	79 seg
campus4-c2	33	+4	19	87,87%	57,57%	79 seg
terrace7-c1	267	+7	139	97,37%	52,05%	200 seg
terrace7-c3	150	+7	70	95,33%	46,66%	200 seg
Promedio	98,8	5,2	49	88,59%	46,04%	127,4 seg

Tabla 4. Valores obtenidos para los videos en ambiente abierto.

De estas tablas podemos obtener también un comportamiento general, visto como el valor promedio de los porcentajes, tanto para los videos en ambiente cerrado (Indoor) y los videos en ambiente abierto (*Outdoor*), siendo los valores de estos porcentajes bastantes cercanos entre sí, con una diferencia de apenas el 5% en los falsos positivos y 0,6% en el porcentaje de ceros, es decir, que tanto en ambientes cerrados y abiertos, se presentó de varias personas que no tenían un valor final de calorías consumidas, aprox. el 46% en ambos casos, así como también tenemos más de un 83% de personas identificadas que en realidad ya estaban presentes en el video.

Con la base de datos tenemos también valores de FPS para cada video, los cuales se encuentran consignados en la Tabla 5 y la Tabla 6, para estos videos la velocidad es bastante buena, superando en su mayoría los 10 FPS en el procesamiento total, esto visto con el reloj interno y no desde *Google Colab*, ya que desde *Colab* aumenta hasta 4 veces o más el tiempo necesario para procesar cada video en línea, estos valores de FPS son incluso mejores que en el video de prueba de Londres, esto a su vez debido a la resolución de la imagen, pues en la base de datos EPFL usamos baja resolución, lo cual representa menos información y por lo tanto, menos gasto computacional, aunque también tiende a ser menos preciso debido a la poca data sobre el sujeto.

Indoor	FPS detección	FPS total
4p-c0	6,8	5,5
4p-c1	13,3	10
6p-c1	12,5	9
pasillo1-c1	15	11
pasillo1-c2	15,2	12,1
Promedio	12,56	9,52

Tabla 5. Rendimiento FPS en base de datos EPFL Indoor

Outdoor	FPS detección	FPS total
campus4-c0	15,2	11
campus4-c1	14,9	11
campus4-c2	13,6	10
terrace7-c1	13,2	9,2
terrace7-c3	12,8	8,5
Promedio	13,94	9,94

Tabla 6. Rendimiento FPS en base de datos EPFL Outdoor.

En relación con el rendimiento en FPS de nuestro algoritmo, lo cual se observa en la Tabla 1, tenemos que decir que el valor real, que observamos nosotros como usuarios desde *Google Colab* es debido a la propia dificultad de las conexiones inalámbricas, puesto que los tiempos de comunicación aumentan (Latencia) y a su vez el tiempo de procesamiento se percibe más largo, ya que en *Google Colab* estamos en constante comunicación con las máquinas virtuales desde las que se ejecuta el programa. Este problema en el *Streaming* de datos disminuye la velocidad en FPS para aplicaciones en tiempo real, es decir, que si ejecutáramos el programa desde nuestro ordenador la mejora sería notable, aunque esto a

su vez depende del poder de nuestra tarjeta gráfica. En el caso de la base de datos EPFL, la tasa de FPS mejoró, esto gracias a la resolución de las imágenes, que disminuye la carga computacional al trabajar con menor cantidad de pixeles.

También sobre este primer video de Londres, las condiciones de lluvia y tiempo del día influyeron notoriamente en la detección de los objetos, pues las intensidades dentro de la imagen confunden al algoritmo, que en principio fue entrenado con una base de datos de imágenes con muy buena iluminación, igualmente en este video la iluminación es muy buena considerando que es de noche, pero los colores oscuros se mimetizan dificultando la diferenciación entre texturas y objetos. La lluvia por su parte favorece esta mimetización, al introducir ruido en la imagen con cada gota de lluvia, se pierde información y no se logra identificar correctamente objetos como por ejemplo, las bicicletas a un costado en la Figura 25, esto influye posteriormente en el seguimiento, pues se pierde de vista el objeto, y por ende, la etiqueta o ID de cada persona cambia, que es lo que ocurre en este video, donde en algunos sujetos el algoritmo supone que se tiene una nueva persona y en el video asigna de forma errónea las etiquetas.

Otro problema presentado con frecuencia y el cual se encuentra constantemente en estudio, es la presencia de falsos positivos, es decir, cuando la red neuronal detecta otros objetos de forma errónea, en nuestro caso, cuando observamos el video de prueba, de la *señora saliendo de su vehículo* (Véase Figura 27), en la parte superior observamos como detecta un buzón en la calle como si fuera otra persona, esto se puede configurar de mejor forma con las cajas de anclaje *o anchor boxes*, ya que colocando el *anchor box* un poco más grande, se favorece a que no detecte objetos tan pequeños como el buzón, pues en el contexto clínico, los sujetos posiblemente no se encuentren tan lejos de la cámara.

También puede llegar a suceder lo contrario, que de hecho no detecte a la persona o no le asigne ninguna etiqueta, lo cual sería un falso negativo, pues en realidad si es una persona pero pasa desapercibida para nuestro algoritmo, uno de las situaciones que favorecen este fenómeno es la oclusión, pues por lo general, se pierde información de la persona que se encuentra tapada en el video resultando en un error para nuestro algoritmo, esto a su vez influye directamente en la perdida de las etiquetas para el seguimiento de personas.

Existe una gran debilidad de este tipo de algoritmos para la detección de objetos, en cuanto a seguridad y precisión, cuando la red neuronal es entrenada se acostumbra a percibir la información de una manera muy precisa, lo cual puede ser afectado con un ataque adversario, ya que este tipo de ataques son diseñados específicamente para provocar un error en el modelo, básicamente son como ilusiones ópticas, pero para computadoras, como ejemplo tenemos la camiseta que fue diseñada para confundir al algoritmo de detección YOLO v2, aquí la persona que lleva puesta la camiseta no la detecta como una persona, aunque claramente es una persona, la camiseta contiene una serie de patrones que para el algoritmo producen un 100% de probabilidad de ser otro objeto como por ejemplo una manzana o un animal, depende simplemente de lo que contenga la camiseta [60]. Este tipo de ataques vulneran la seguridad para utilizar algoritmos de inteligencia artificial en varios campos, aunque son específicamente diseñados, es decir, no se presentan en la naturaleza, son un tema que aún se sigue investigando para desarrollar mecanismos de defensa.

Ahora bien, respecto al uso de la cámara en tiempo real, esto se puede hacer trabajando desde un ordenador con programas como Anaconda, visual Studio y similares, donde se puede ejecutar código en Python, dado que el algoritmo que vamos utilizado está en dicho lenguaje de programación. Pero en nuestro caso, al trabajar desde máquinas virtuales de *Google Colab*, no contamos con puertos de entrada, ya que estamos trabajando desde la nube, y no existe tal cosa como un ratón virtual, teclado o cámara web que se puedan utilizar desde dicha máquina virtual, es decir, físicamente estamos usando una caja negra con el hardware o GPU que Google nos facilita para hacer trabajos con IA. Aunque existen programas o algoritmos en *Google Colab* que utilizan java para establecer una conexión entre la cámara web de nuestro ordenador y la máquina virtual, para aplicaciones en tiempo real no resulta sensato, puesto que presenta retrasos notorios debido a la misma latencia de hacer *stream* y procesar la detección para cada fotograma. Pues secuencialmente se envía el fotograma actual de la cámara web hacia la máquina virtual, se procesa para obtener los cuadros delimitadores y se envían de vuelta a nuestro navegador para ser sobrescrito en la imagen actual, un proceso bastante tardado. [61]

Con lo anterior, estamos perdiendo en gran parte la ventaja de la tarjeta gráfica GPU, que, en comparación al procesador del computador o CPU, realiza el procesamiento de información de forma paralela utilizando su gran cantidad de núcleos de procesamiento CUDA, que, aunque son menos potentes comparados con la unidad de procesamiento de un CPU, son específicos para trabajar imágenes y procesos matemáticos más simples, permitiendo realizar múltiples cálculos en el mismo instante.

Finalmente, con nuestros resultados clínicos, como lo son las gráficas y resultados de velocidad, calorías y distancia se tiene una estimación de la cuantificación, pues igualmente el método de cambio de perspectiva está basado en una calibración bidimensional debido a que los videos son obtenidos con cámaras monoculares, por lo cual no posee gran percepción del espacio 3D, estos valores pueden indicar lo ocurrido en el video aunque no tengan gran precisión, pues como se observa en los valores de calorías de la Tabla 2, son resultados acordes a lo ocurrido, pues por ejemplo, en ese video de los *sospechosos de lexington*, tenemos que el sujeto 1 recorre 12 m y el sujeto 2 recorre 6 metros, esto a los ojos del espectador es lo que ocurre, pues en el video, debido a la oclusión pareciera que un sujeto recorre el doble de distancia que el otro. Ya en la Figura 28 vemos la velocidad, que a lo largo del video si hace sentido del movimiento de los sospechosos, pues cuando se quedan quietos, este valor se acerca a cero. En los mapas de calor y trayectoria de la Figura 29, vemos que los sujetos del video EPFL se desplazan en un mismo sentido, con una salida común de la estación de metro.

Las anteriores variables sumadas a otros datos de monitoreo del paciente, como lo son saturación de oxígeno, frecuencia cardiaca y frecuencia pulmonar pueden ayudar a tener un control absoluto de las variables dinámicas del entrenamiento, para llevar un control y registro que permita obtener una mejora en la condición física del paciente. Tal como se explica en la sección “Análisis de Resultados Clínicos” del capítulo anterior.

Capítulo 7 – Conclusiones

Se pudo diseñar un algoritmo que cumple con los requisitos planteados en un comienzo, que, aunque se puede mejorar en bastantes aspectos, el principal bloqueo que encontramos es el propio algoritmo de detección y seguimiento de personas, en cuyo caso con un sistema más robusto se pueden obtener mejores resultados, además, en aplicaciones de este estilo puede resultar útil usar imágenes de alta resolución, aunque disminuye la tasa de FPS procesada.

Del trabajo dirigido podemos concluir que la detección y seguimiento de personas es un ámbito que debe seguir en investigación, debemos reconocer que la principal dificultad para este tipo de análisis de video es la oclusión, la cual afecta directamente el resultado del estudio y que para un sistema monocular presenta aún más importancia, ya que tal y como se mencionó en la sección de recomendaciones existen algunas alternativas más complejas que pueden llegar a reducir este error.

Uno de los principales problemas en el desarrollo del algoritmo es la calibración del video y la transformación de perspectiva, ya que en este trabajo dirigido no se contaban con medios físicos para reunirse debido a la contingencia producto del COVID 19, de forma que se pudiera trabajar otras alternativas con videos propios con cámaras calibradas o más complejas, la aproximación para calcular las distancias es bastante simple y empírica, lo que se propone en la sección anterior debe ser completamente obligatorio para realizar trabajos de campo con Inteligencia artificial en el diagnóstico y seguimiento de pacientes en la vida real, ya que si en un principio los valores de distancia recorrida se encuentran mal, los valores dependientes de él también se verán afectados.

Para el análisis en tiempo real, resulta imprescindible utilizar una maquina local, ya que con máquinas virtuales, la latencia juega un papel en contra, dificultando obtener velocidades de procesamiento útiles para este tipo de aplicaciones, pues se pierde en gran parte la característica principal de las Redes neuronales artificiales, que consiste en realizar trabajos de forma paralela, pues *Colab* ejecuta el programa de forma secuencial, tanto su entorno como para la obtención de los gráficos y resultados.

En el entorno clínico los métodos basados en IA pueden resultar bastantes complejos para su entendimiento, ya que la IA tiene una naturaleza de “caja negra”, ya que el algoritmo llega a conclusiones mediante procesos matemáticos tan complejos que no pueden ser comprendidos por el ser humano, así como los ataques adversarios, existen problemas que no pueden ser percibidos a primera vista por el ser humano, por ello debe existir una regulación internacional que controle y estandarice estos métodos médicos basados en IA, creando leyes y regulaciones que disminuyan estos riesgos que no se guían por lo tradicional. [62]

Capítulo 8 - Recomendaciones y trabajos futuros

En esta sección se realizan algunos comentarios respecto a las dificultades presentadas para el desarrollo del trabajo dirigido, con el fin de ofrecer una retroalimentación positiva y mejorar el desarrollo o funcionamiento del algoritmo para la detección y seguimiento de personas.

Uno de los problemas más recurrentes para la detección de personas es el efecto de la oclusión visual, como se comentó en otras secciones este problema es inevitable en una cámara con lente monocular, por lo tanto, como mejora a este inconveniente puede emplearse una disposición de cámaras dispuestas al reconocimiento de objetos desde diferentes ángulos de visión, como es el caso de algunos estudios de biomecánica donde utilizan sensores y cámaras en diferentes lugares dentro de la habitación, esto nos puede favorecer el estudio y la eficiencia del algoritmo para realizar el seguimiento de personas. Ya que con varias cámaras es posible identificar directamente la posición de las personas en una vista diferente, mejorando el cálculo de las trayectorias de los sujetos, y evitando el uso de software como seria DeepSORT, ya que no sería necesario predecir la posición del sujeto en el fotograma siguiente, sino que tendría la referencia o el valor desde la cámara en donde no se presenta oclusión. Ahora bien, este sistema podría presentar a su vez el mismo inconveniente, si por ejemplo tenemos oclusión también desde la cámara adicional, en cuyo caso a medida se tenga un mayor número de cámaras puede disminuirse también este problema, la configuración ideal de las cámaras debe ser en lados contrarios u ortogonales según se considere, para ofrecer cambios significativos en la imagen.

Otra opción referente al problema de conocer la posición exacta del sujeto en el video, es utilizar cámaras con sensores de proximidad, de forma que se tenga un valor real de medición de la distancia que recorren las personas, con esta tecnología es posible dibujar un video 2D que de forma intrínseca contenga la información de posición a modo de mapa de calor, como por ejemplo con la cámara OAK—D basada en *openCV* y optimizada para trabajar con redes neuronales profundas, especialmente para el seguimiento de objetos, detección de gestos y movimiento de personas en un entorno enfocado a inteligencia artificial espacial. [63]

Adicional a lo anterior, respecto a trabajos futuros se prevé que la inteligencia artificial siga en constante evolución, llegando a desarrollar labores importantes en sectores de salud, industria y comercio, aunque esto a su vez presenta un riesgo en la población mundial, el cual debe ser analizado con el fin de determinar el posible impacto social, económico y cultural, ya que ha sido por estos motivos que varios programas de inteligencia artificial no salen a la luz, puesto que de forma análoga al cerebro, puede tener una forma de actuar en pro de la humanidad o en contra, como el caso del autor original de YOLOv3, Joseph Chet Redmon quien observó como su trabajo estaba siendo empleado para la elaboración de armamento militar, motivo por el cual dejo la investigación y su equipo de trabajo de desarrolladores.

Para trabajos futuros se puede considerar trabajar con YOLO v4, o YOLO v5 si se encuentra una versión oficial, pues, aunque YOLO v3 es robusto y preciso, tal como se prevé, se puede lograr mayor rendimiento con estos algoritmos junto a DeepSORT, de forma que se obtenga un video con mejor tasa de FPS, detección e identificación de personas.

Referencias

[1] A. Mercadé, «Los 8 tipos de inteligencia según Howard Gardner: la teoría de las inteligencias múltiples.,» 2019.

[2] I. Gonzales, «Armas que disparan solas: así funcionan las ametralladoras con inteligencia artificial,» *El español,* 29 12 2020.

[3] G. P. RAMIREZ, «LA INTELIGENCIA ARTIFICIAL Y SU INFLUENCIA EN LA EFICIENCIA DEL COMERCIO INTERNACIONAL,» Lima-Perú, 2020.

[4] I. Montenegro Trujillo, A. Hernandez, D. Chavarro, M. I. Velez, G. Tovar, A. M. Niño y A. Olaya, «MACROTENDENCIAS HACIA EL 2030: El mundo y América Latina,» *Colciencias,* nº 2, 2018.

[5] «Lee, KF. “La Inteligencia Artificial Y El Futuro Del Trabajo: Una Perspectiva China” Openmind. Artículo de investigación (2020)».

[6] «Sreenu, G., Saleem Durai, M.A. Intelligent video surveillance: a review through deep learning techniques for crowd analysis. J Big Data 6, 48 (2019).».

[7] «Ejercicio terapéutico,» ASEPEYO, 03 09 2018. [En línea]. Available: https://salud.asepeyo.es/profesionales/rehabilitacion/ejercicio-terapeutico/. [Último acceso: 24 05 2021].

[8] «S.Saranya, Dr. P. JesuJayarin “A Survey For Tracking And Monitoring The Alzheimer Patient” - IEEE Conference Publication. (2017)».

[9] Y. S. Gutiérrez Pardo y A. M. Guerrero Carhuavilca, «Seguimiento del Paciente en Rehabilitación con Kinect 2.0,» UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS, Lima, 2019.

[10] «Inteligencia artificial,» Wikipedia, [En línea]. Available: https://es.wikipedia.org/wiki/Inteligencia_artificial. [Último acceso: 2021 04 11].

[11] «Las profesiones que podrían desaparecer por la Inteligencia Artificial,» *El tiempo,* 2 enero 2021.

[12] B. Martín, «Cómo crear inteligencia artificial que no exhiba prejuicios humanos,» BBVA Open Mind, 13 noviembre 2019. [En línea]. Available: https://www.bbvaopenmind.com/tecnologia/inteligencia-artificial/como-crear-inteligencia-artificial-que-no-exhiba-prejuicios-humanos/. [Último acceso: 16 marzo 2021].

[13] C. Saavedra y F. Izaurieta, «Redes Neuronales Artificiales,» Concepcion, Chile, 2000.

[14] J. M. Fernandez Fernandez y R. Florez Lopez, Las redes neuronales artificiales, La Coreña, España: NetBiblo, 2008.

[15] «Wikipedia,» 16 febrero 2021. [En línea]. Available: https://es.wikipedia.org/wiki/Aprendizaje_autom%C3%A1tico. [Último acceso: 16 marzo 2021].

[16] «Sobreajuste,» Wikipedia, 11 octubre 2020. [En línea]. Available: https://es.wikipedia.org/wiki/Sobreajuste. [Último acceso: 16 marzo 2021].

[17] «Aprendizaje profundo,» Wikipedia, 01 03 2021. [En línea]. Available: https://es.wikipedia.org/wiki/Aprendizaje_profundo. [Último acceso: 24 05 2021].

[18] «Te damos la bienvenida a Colaboratory,» Google, [En línea]. Available: https://colab.research.google.com/notebooks/intro.ipynb?hl=es. [Último acceso: 16 marzo 2021].

[19] Google, «TensorFlow,» [En línea]. Available: https://www.tensorflow.org/?hl=es-419. [Último acceso: 15 04 2021].

[20] «RFID,» Wikipedia, [En línea]. Available: https://es.wikipedia.org/wiki/RFID. [Último acceso: 24 05 2021].

[21] «Bluetooth de baja energía,» Wikipedia, [En línea]. Available: https://es.wikipedia.org/wiki/Bluetooth_de_baja_energ%C3%ADa. [Último acceso: 24 05 2021].

[22] «NFC,» Wikipedia, [En línea]. Available: https://es.wikipedia.org/wiki/Near_field_communication. [Último acceso: 24 05 2021].

[23] «GPS,» Wikipedia, [En línea]. Available: https://es.wikipedia.org/wiki/GPS. [Último acceso: 24 05 2021].

[24] F. S. Caparrini, «Entrenamiento de Redes Neuronales: mejorando el Gradiente Descendente,» Dpto. Ciencias de la Computación e Inteligencia Artificial, Universidad de Sevilla., [En línea]. Available: http://www.cs.us.es/~fsancho/?e=165. [Último acceso: 24 05 2021].

[25] K. O'Shea y R. Nash, «An Introduction to Convolutional Neural Networks,» arXiv, Gales, UK, 2015.

[26] S. Sharma y A. Athaiya, «ACTIVATION FUNCTIONS IN NEURAL,» International Journal of Engineering Applied Sciences and Technology, 2020.

[27] F. f. Li, J. Johnson y S. Yeung, «Lecture 4: Backpropagation and Neural Networks,» 13 04 2017. [En línea]. Available: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf. [Último acceso: 24 05 2021].

[28] «Red convolucional en PyTorch,» Cleverpy Machine Learning, 09 01 2018. [En línea]. Available: https://medium.com/@cleverpysolutions/red-convolucional-en-pytorch-c499ae8af6ca. [Último acceso: 24 05 2021].

[29] «Max-pooling / Pooling,» 2018. [En línea]. Available: https://computersciencewiki.org/index.php/Max-pooling_/_Pooling. [Último acceso: 24 05 2021].

[30] C.-W. Zhang, M.-Y. Yang y H.-J. Zeng, «Pedestrian detection based on improved LeNet-5 convolutional neural network,» Journal of Algorithms & Computational Technology, shaanxi, China, 2019.

[31] J. Huang, V. Rathod, C. Sun , M. Zhu y S. Guadarrama, «Speed/accuracy trade-offs for modern convolutional object detector,» Actas de la Conferencia IEEE sobre visión artificial y reconocimiento de patrones, 2017.

[32] J. U. T. G. A. S. T. Gevers, «Selective Search for Object Recognition,» Italia y Amsterdam, 2012.

[33] R. Gandhi, «R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms,» Towards Dara Science, [En línea]. Available: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e. [Último acceso: 17 04 2021].

[34] R. Girshick, «Fast r-cnn. In Proceedings of the IEEE international conference on computer vision,» Microsoft Research, 2015.

[35] K. He, G. Gkioxari, P. Dollar y R. Girshick, «Mask R-CNN,» Facebook AI Research.

[36] A. Garcia Garcia, S. Oprea, S. Orts Escolano, J. Garcia Rodriguez y V. Villena Martinez, «A Review on Deep Learning Techniques Applied to Semantic Segmentation,» arXiv, 2017.

[37] X. Feng, Y. Jiang, X. Li, X. Yang y M. Du, «Computer vision algorithms and hardware implementations: A survey,» ELSEVIER, Shanghai, 2019.

[38] J. Redmon, S. Divvala, R. Girshick y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection,» 2015.

[39] A. Persson, «Non Max Suppression Explained and PyTorch Implementation,» Youtube, 07 10 2020. [En línea]. Available: https://www.youtube.com/watch?v=YDkjWEN8jNA. [Último acceso: 24 05 2021].

[40] D. Patel, «What is YOLO algorithm? | Deep Learning Tutorial 31 (Tensorflow, Keras & Python),» 25 12 2020. [En línea]. Available: https://www.youtube.com/watch?v=ag3DLKsl2vk. [Último acceso: 24 05 2021].

[41] J. Redmon y A. Farhadi, «YOLO9000: Better, Faster, Stronger,» 2016.

[42] A. Kathuria, «What's new in YOLO v3?,» towards datascience, 23 04 2018. [En línea]. Available: https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b. [Último acceso: 24 04 2021].

[43] J. Redmon y A. Farhadi, «YOLOv3: An Incremental Improvement,» Washington, 2018.

[44] A. Bochkovskiy, C.-Y. Wang y H. Y. Mark Liao, «YOLOv4: Optimal Speed and Accuracy of Object Detection,» Taiwan, 2020.

[45] R. Balsys, «YOLO v3 Real-Time Object tracking with Deep SORT,» 26 Junio 2020. [En línea]. Available: https://pylessons.com/YOLOv3-TF2-DeepSort/. [Último acceso: 16 Marzo 2021].

[46] S. DEY, «Implementing Lucas-Kanade Optical Flow algorithm in Python,» 25 02 2018. [En línea]. Available: https://sandipanweb.wordpress.com/2018/02/25/implementing-lucas-kanade-optical-flow-algorithm-in-python/. [Último acceso: 23 05 2021].

[47] S. R. Maiya, «DeepSORT: Deep Learning to Track Custom Objects in a Video,» 2019. [En línea]. Available: https://nanonets.com/blog/object-tracking-deepsort/. [Último acceso: 24 04 2021].

[48] A. Bewley, Z. Ge, L. Ott, B. Upcrof y F. Ramos, «SIMPLE ONLINE AND REALTIME TRACKING,» Sindney, 2016.

[49] N. Wojke, A. Bewley y D. Paulus, «SIMPLE ONLINE AND REALTIME TRACKING WITH A DEEP ASSOCIATION METRIC,» Brisbane, 2017.

[50] f. fleuret, j. berclaz y p. fua, «Multi-camera pedestrians video,» EPFL, [En línea]. Available: https://www.epfl.ch/labs/cvlab/data/data-pom-index-php/. [Último acceso: 25 04 2021].

[51] «Walking London's SOHO in HEAVY RAIN - Saturday Evening City Ambience,» Watched Walker, 27 07 2020. [En línea]. Available: https://www.youtube.com/watch?v=__Eo-dvEH7g. [Último acceso: 26 04 2021].

[52] «10 WEIRD THINGS CAUGHT ON SECURITY CAMERAS & CCTV,» CREATIVE, 28 03 2020. [En línea]. Available: https://www.youtube.com/watch?v=sPwxPxgJDSc. [Último acceso: 26 04 2021].

[53] «Two people walk up to home, walk away after seeing security camera,» Lexington Police Department, 23 01 2019. [En línea]. Available: https://www.youtube.com/watch?v=bFrwLUjXcLs. [Último acceso: 26 04 2021].

[54] A. Augustyn, «Why Are Basketball Hoops 10 Feet High?».*Britannica.*

[55] A. L. González Gómez, J. E. Meneses Fonseca, A. Ballesteros Díaz y J. León Téllez, «Método para la extracción de puntos de control en la calibración de una cámara basado en la estimación de la fase de un patrón de puntos codificado,» Scielo, Bucaramanga, 2013.

[56] «¿Dónde están los hombres y mujeres más altos de Suramérica?,» *EL TIEMPO Casa Editorial,* 29 06 2016.

[57] S. G. MATIAS, «Colombia, uno de los países con más dificultades en acceso a internet,» *EL TIEMPO editorial,* 11 09 2020.

[58] S. C. Walpole, D. Prieto Merino, J. Cleland, P. Edwards y G. Stevens, «The weight of nations: an estimation of adult human biomass,» BMC Public Health , London, 2012.

[59] S.-S. Wu y H.-Y. Wu, «The Design of an Intelligent Pedometer using Android,» Taiwan, 2011.

[60] K. Xu, G. Zhang y H. Chen, «Adversarial T-shirt! Evading Person Detectors in A Physical World,» IBM Research, Massachusetts, 2020.

[61] T. I. guy, «Google Colab: Access Webcam for Images and Video,» [En línea]. Available: https://colab.research.google.com/drive/1QnC7lV7oVFk5OZCm75fqbLAfD9qBy9bw?usp=sharing. [Último acceso: 25 04 2021].

[62] C. Silcox, «La inteligencia artificial en el sector salud,» Banco Interamericano de Desarrollo. , 2020.

[63] OpenCV, «Spatial AI powerhouse, OAK-D,» [En línea]. Available: https://store.opencv.ai/. [Último acceso: 15 04 2021].