

INSTITUTO TECNOLÓGICO DE BUENOS AIRES



Algoritmo basado en inteligencia artificial para medición de frecuencia cardíaca a través de videos de dispositivos de uso personal

TRABAJO FINAL PRESENTADO PARA LA OBTENCIÓN DEL TÍTULO
DE BIOINGENIERÍA

Nicole Bartellini Huapalla - 56138
Martina Bunge - 58638

Dr. Gustavo Daquarti
Director

Üma Health
Institución

18 de Septiembre de 2023

Índice de contenidos

Índice de contenidos	ii
Índice de figuras	v
Índice de tablas	viii
Glosario	x
Resumen	xi
1. Introducción y motivaciones	1
1.1. ÜMA	4
1.2. Estado del arte	5
1.3. Objetivos	7
2. Marco teórico	8
2.1. Fotopletismografía	8
2.2. Fotopletismografía remota	11
2.3. Inteligencia Artificial y salud	12
2.4. Machine Learning y Deep Learning	13
2.5. Redes Neuronales Artificiales	15
2.5.1. Perceptrón simple	16
2.5.2. Funciones de activación	18
2.5.3. Perceptrón multicapa	19
2.5.4. Redes Neuronales Convolucionales	20
2.5.5. Capas convolucionales	21
2.5.6. Capas de pooling	24
2.5.7. Sobreajuste	25
2.5.8. Capas densas o fully-connected	26
2.5.9. Funciones de pérdida	26
2.5.10. Optimizadores y back-propagation	28
2.5.11. Redes convolucionales 2D	29

2.5.12. Redes convolucionales 3D	30
2.6. Evaluación del modelo	31
2.6.1. Raíz del error cuadrático medio (RMSE)	31
2.6.2. Error absoluto medio (MAE)	31
2.6.3. Coeficiente de Pearson (ρ)	31
2.6.4. Coeficiente de Determinación (R^2)	32
3. Materiales y métodos	33
3.1. Lenguaje de programación	33
3.2. Enfoques	34
3.3. Datasets empleados	35
3.3.1. Datasets generados en Üma Health	36
3.3.2. Dataset público	42
3.4. Métodos aplicados a los videos obtenidos del dedo índice de los sujetos	43
3.4.1. Preparación de datasets	44
3.4.2. Obtención de señal de FPG	44
3.4.3. Entrenamiento de un primer modelo de predicción de frecuencia cardíaca	44
3.4.4. Análisis de señales de FPG	47
3.4.5. Entrenamiento de una red convolucional 2D clasificadora de señales	49
3.4.6. Preprocesamiento de señales	52
3.4.7. Entrenamiento de modelos de predicción de frecuencia cardíaca	54
3.4.8. Métodos de evaluación	55
3.5. Métodos aplicados a los videos obtenidos de la cara de los sujetos	56
3.5.1. Preparación de los datasets	56
3.5.2. Selección del área de interés	57
3.5.3. Magnificación de la señal	60
3.5.4. Entrenamiento de modelos para predicción de la frecuencia cardíaca	61
3.5.5. Postprocesamiento de la señal de fotopletismografía como output	69
3.5.6. Métodos de evaluación	69
4. Resultados y discusión	72
4.1. Métodos aplicados a los videos obtenidos del dedo índice de los sujetos.	72
4.1.1. Primeros modelos de predicción de frecuencia cardíaca	72
4.1.2. Modelo clasificador de señales	74
4.1.3. Modelo de predicción de la frecuencia cardíaca	75
4.2. Métodos aplicados a los videos obtenidos de la cara de los sujetos	77
4.2.1. Modelos de predicción de frecuencia cardíaca como valor numérico escalar	77

4.2.2. Modelos de predicción de la señal de fotopletismografía	80
4.3. Modelos seleccionados	84
4.4. Desafíos futuros	85
Conclusiones	87
A. Tablas y figuras anexas	89
A.1. Estructuras de redes convolucionales para los modelos de fotopletismografía de contacto	89
A.2. Tablas de resultados para los modelos de fotopletismografía remota con output de valor escalar	92
A.3. Gráficos de resultados	94
Bibliografía	100

Índice de figuras

1.1.	Densidad de médicos, enfermeros, parteras, dentistas, farmacéuticos por 10000 habitantes, según la región de la ONU, en 2022 [1].	2
1.2.	Porcentaje de hogares con acceso a computadora e internet, según región argentina en el cuarto trimestre de 2021[2].	3
1.3.	Accesos a internet fijos y móviles. Enero 2015- junio 2022, Argentina[3].	4
2.1.	Arreglo de los sensores y LED en fotopletismografías. A la derecha, por reflexión. A la izquierda, por transmisión [4].	9
2.2.	Componentes de la señal de fotopletismografía [4].	10
2.3.	Diagrama de una red neuronal artificial convencional con 3 capas ocultas, una capa de entrada y una capa de salida. Los pesos que caracterizan las uniones entre la capa j y la capa anterior i son denominados w_{ij} [5].	17
2.4.	Diagrama de un perceptrón simple y sus componentes. Hay una única neurona con n inputs $x_1, x_2, x_3, x_4, \dots, x_n$. Estas entradas son multiplicadas por sus pesos respectivos ($w_1, w_2, w_3, w_4, \dots, w_n$) y sumado el bias b correspondiente.	17
2.5.	Diagrama de un perceptrón multicapa y su estructura. [6]	20
2.6.	Diagrama de proceso de convolución [7]. Se toma una imagen de un solo canal representada por una matriz de 8x8 (matriz azul) con un kernel de 3x3 (verde).	23
2.7.	Convolución con zero-padding. De izquierda a derecha: imagen de entrada, kernel y matriz de activación [8].	24
2.8.	Ejemplos de tipos de pooling [9]. Se tiene una matriz de entrada de 4 x 4 y el pooling usará un kernel de 2 x 2. Por lo tanto, la matriz de salida tendrá dimensiones de 2 x 2, en el caso de average y max pooling. En el avergae pooling, cada valor de la salida será un promedio de los valores de la ventana; mientras que con el max pooling, el valor final será el máximo de aquellos que se encuentren en la ventana.	25
2.9.	Modificaciones dimensionales del vector output tras pasar por capa densa y flattening [7].	27

2.10. Convolución de 3 dimensiones aplicada a videos [10].	30
3.1. Diagramas de medición de las muestras. (1) En la mano derecha del sujeto se coloca el dedo cubriendo la cámara y el flash [11], (2) se coloca la cámara a cierta distancia del sujeto de forma que se encuadre el rostro [11] y (3) en la mano izquierda se coloca el oxímetro de pulso comercial para obtener las etiquetas.	37
3.2. Características de la población que compone el dataset UMA-rPPG y UMA-dPPG: (a) Sexo y (b) Barba que oculte parte del rostro.	41
3.3. Estructura de carpetas dentro de Cloud storage para los datasets del proyecto.	42
3.4. Imágenes de videos del dedo cubriendo la cámara tomados con dos dispositivos distintos. En el cuadro del video (a), la mayor iluminación se encuentra en la porción superior, mientras que en el video (b) la porción iluminada se encuentra en la esquina inferior izquierda.	45
3.5. Proceso de obtención de tres señales candidatas a FPG a partir de un video.	45
3.6. Señales obtenidas con 3 videos pertenecientes al dataset Uma-rPPG . .	47
3.7. señales obtenidas con tres videos pertenecientes al dataset Uma-rPPG-extensión	48
3.8. Arquitectura de red VGG.	50
3.9. Arquitectura de red ResNet.	51
3.10. Proceso de selección de mejor señal de FPG correspondiente a un video utilizando un modelo clasificador de señales	53
3.11. Efecto de filtro Butterworth de orden 2 en señal de video FPG	54
3.12. Flujo de preprocesamiento de los videos.	56
3.13. Patrones Haar para detección de rostros [12].	58
3.14. Haar-like features [13].	58
3.15. Magnificación del ruido generado por el parpadeo.	61
3.16. Diagrama de flujo de entrenamiento de la red tridimensional con output de fotopletismografía.	68
3.17. Postprocesamiento y comparación de los modelos	69
4.1. Frecuencias cardíacas predichas por modelo 1DCNN-v1.3 en función de las frecuencias cardíacas reales	73
4.2. Frecuencias cardíacas predichas por modelo 1DCNNv2.4 en función de las frecuencias cardíacas reales	76
4.3. Distribución de frecuencias cardíacas medidas por el oxímetro en el dataset de entrenamiento (en azul) y testeо (en cian) de los modelos entrenados con videos del dedo	77

4.4. Distribución de frecuencias cardíacas medidas por el oxímetro en el dataset de entrenamiento (en azul) y validación (en cian) de los modelos con <i>output</i> de valor escalar.	78
4.5. Distribución de frecuencias cardíacas (a) calculadas a partir de los FPGs originales en el dataset de entrenamiento y validación de los modelos con <i>output</i> de señal; (b) medidas con un oxímetro de pulso para el dataset de Uma-rPPG.	80
4.6. Mejores 3 y peores 3 resultados de la predicción de la señal de fotopletismografía en comparación a la señal original.	81
4.7. Frecuencia cardíaca estimada a partir de la señal predicción en función de las frecuencias estimadas a partir de la señal original.	82
4.8. Frecuencia cardíaca estimada a partir de la señal predicción en función de las frecuencias obtenidas a partir del oxímetro comercial. Los valores con marcadores naranjas corresponden a los valores que se descartaron para un análisis más profundo de los resultados.	83
4.9. MAE (a) y RMSE (b) en función de las frecuencias obtenidas a partir del oxímetro comercial.	84
A.1. Frecuencias cardíacas predichas por los modelos con <i>output</i> valor escalar en función de las frecuencias cardíacas obtenidas del oxímetro en el dataset de validación.	95
A.2. Frecuencias cardíacas predichas por los modelos con <i>output</i> valor escalar en función de las frecuencias cardíacas obtenidas del oxímetro en el dataset de testeo.	97
A.3. Frecuencias cardíacas predichas por los modelos con <i>output</i> valor escalar en función de las frecuencias cardíacas obtenidas del oxímetro en el dataset de testeo limitado a valores menores a 100 lpm.	99

Índice de tablas

1.1.	Desempeño de la estimación de la frecuencia cardíaca de los desarrollos relevantes a ser comparados con el modelo final con dataset públicos [14].	7
3.1.	Especificaciones de los datasets utilizados en el desarrollo del algoritmo relacionados a los modelos en los que fueron utilizados.	35
3.2.	Estructura de la red convolucional 1D correspondiente a los modelos 1DCNNv1.x	46
3.3.	Parametros utilizados para el entrenamiento de cada red neuronal convolucional 1d	47
3.4.	Estructura de la red convolucional 1D correspondiente al modelo v4 . .	54
3.5.	Hiperparámetros de arquitectura y entrenamiento de cada modelo . . .	55
3.6.	Estructura de la red convolucional 3D con output de frecuencia cardíaca como valor numérico.	62
3.7.	Parámetros de entrenamiento de los diversos modelos.	64
3.8.	Parámetros de las capas de la red convolucional 3D con output de frecuencia cardíaca como valor numérico.	65
3.9.	Estructura de la red convolucional 3D con output de señal de FPG. . .	66
3.10.	Parámetros de las capas de la red convolucional 3D con output de señal de fotopletismografía.	68
4.1.	Metricas de entrenamiento y validación de primeros modelos convolucionales 1D entrenados	72
4.2.	Metricas de entrenamiento y validación de primeros modelos convolucionales 1D entrenados	73
4.3.	Métricas de validación de los modelos clasificadores de señales	74
4.4.	Métricas de los modelos con <i>output</i> de frecuencia cardíaca para los datos de entrenamiento y validación con la toma del dedo índice: MAE y RMSE entre la frecuencia original y la predicción.	75
4.5.	Métricas de los modelos a comparar a través de la frecuencia cardíaca de la predicción contra los datos originales.	83

A.1. Estructura de la red convolucional 1D correspondiente al modelo 1DCNNv2.1	89
A.2. Estructura de la red convolucional 1D correspondiente al modelo 1DCNNv2.2	90
A.3. Estructura de la red convolucional 1D correspondiente a los modelos 1DCNNv2.3 y 1DCNNv2.6	90
A.4. Estructura de la red convolucional 1D correspondiente al modelo 1DCNNv2.6	91
A.5. Métricas de los modelos en entrenamiento con <i>output</i> de frecuencia cardíaca: MAE y RMSE.	92
A.6. Métricas de los modelos en validación con <i>output</i> de frecuencia cardíaca: MAE, RMSE y correlación entre la frecuencia cardíaca predicha y la frecuencia medida por el oxímetro.	92
A.7. Métricas de los modelos en Test y Test acotado con <i>output</i> de frecuencia cardíaca: MAE, RMSE y correlación entre la frecuencia cardíaca predi- cha y la frecuencia medida por el oxímetro.	93

Glosario

ANN: redes neuronales artificiales, por sus siglas en inglés *Artificial Neural Networks*.

CNN: redes neuronales convolucionales, por sus siglas en inglés *Convolutional Neural Network*.

DNN: redes neuronales profundas, por sus siglas en inglés *Deep Neural Network*.

FC: capa densa o *fully-connected*.

FPG: fotopletismografía.

fps: frames por segundo, medida de la frecuencia de muestreo en un video.

GPU: unidad gráfica de procesamiento.

IA: inteligencia artificial.

INDEC: Instituto Nacional de Estadística y Censos.

LSTM: red recurrente de memoria a corto plazo, por sus siglas en inglés *Long short-term memory*.

lpm: latidos por minuto.

MAE: error absoluto medio.

MLP: Perceptrón multicapa, por sus siglas en inglés *Multi-layer Perceptron*.

MSE: error cuadrático medio.

nm: nanómetros, unidad de longitud del Sistema Internacional de Unidades que equivale a una mil millonésima parte de un metro.

O₂Hb: oxihemoglobina.

OMS: Organización Mundial de la Salud.

ONU: Organización de las Naciones Unidas.

px: píxel, unidad mínima que compone a una imagen.

R₂: coeficiente de determinación.

rFPG: fotopletismografía remota.

RGB: canales de sensores de luz rojo, verde y azul.

RHb: deoxihemoglobina.

ρ: coeficiente de la correlación de Pearson.

RMSE: raíz del error cuadrático medio.

ROI: región de interés de una imagen, por sus siglas en inglés *Region of Interest*.

SD: desvío estándar

Resumen

La teleconsulta mediante dispositivos móviles ha demostrado ser útil en el diagnóstico y monitoreo de los pacientes; sin embargo, la imposibilidad de realizar un examen físico completo limita sus beneficios y adopción. Las alteraciones de frecuencia cardíaca son un indicador de potenciales patologías, cuya rápida detección ayuda tanto a la prevención como al seguimiento del paciente y, teóricamente, la medición de la misma podría ser estimada a través de fotopletismografías utilizando la cámara de dispositivos comerciales. Los modelos de Deep Learning han demostrado resultados prometedores en el procesamiento de señales y es factible su implementación para extraer las señales de fotopletismografías.

El objetivo de este proyecto fue generar un algoritmo capaz de extraer la frecuencia cardíaca a partir de vídeos de distinto origen (dedo índice y rostro del sujeto) obtenidos por dispositivos de uso personal tales como smartphones y computadoras utilizando redes convolucionales y otros procesamientos de imágenes con enfoques más tradicionales. Con esta herramienta, se propone garantizar el acceso a parámetros fisiológicos que describan la situación del paciente de manera remota sin necesidad de instrumentación específica.

Se utilizaron dos bases de datos para el entrenamiento del modelo de inteligencia artificial. La primera está compuesta por videos de la cara (UMA-rPPG) y del dedo índice (UMA-dPPG) del sujeto asociados a una frecuencia cardíaca obtenida de un oxímetro de pulso; y la segunda base de datos utilizada fue UBFC, un dataset público y accesible en internet que contiene videos de la cara asociados a señales de fotopletismografía de las que se pueden estimar la frecuencia cardíaca.

Los datos de UMA-dPPG y UMA-rPPG fueron utilizados para entrenar los modelos de obtención de la frecuencia cardíaca como valor numérico a partir un video del dedo índice y a partir un video de la cara, respectivamente. Se analizaron diferentes alternativas de preprocesamiento para mejorar no sólo el rendimiento general del modelo frente a las predicciones de la frecuencia cardíaca, sino también su velocidad y eficiencia en situaciones de uso reales. Para analizar y comparar los resultados de cada modelo se utilizó el error medio absoluto (MAE), la raíz del error cuadrático medio (RMSE) y la correlación de Pearson entre las frecuencia predichas y las frecuencias etiquetadas. Adicionalmente, con el dataset de UBFC, se generó un modelo de predicción de señales

de fotopletismografías y fue evaluada la similitud entre la señal predicción y la señal original con la correlación de Pearson.

En primera instancia, con la partición de validación del dataset UMA-dPPG, el mejor modelo entrenado obtuvo 11,08 latidos por minuto (lpm) y 7,58 lpm para RMSE y MAE, respectivamente. Dicho modelo fue testeado con nuevo dataset recolectado. En dicho test, el RMSE fue de 10,01 lpm y el MAE, 7,02 lpm. Se calculó la correlación entre las frecuencias predichas y las originales obteniendo como resultado 0,84.

Para los modelos con vídeos de la cara en la partición de validación, la correlación de Pearson del modelo elegido fue de 0,82 comparando la señal de fotopletismografía original y la señal-predicción; un RMSE de 4,12 lpm y MAE de 2,43 lpm entre las frecuencias calculadas de cada señal. Sin embargo, al probar el modelo con un dataset de otro dominio, UMA-rPPG, los resultados de RMSE y MAE fueron 15,89 y 13,08 lpm respectivamente.

Este trabajo muestra que la estimación de la frecuencia cardíaca es factible mediante la utilización de cámaras comerciales de smartphones y computadoras a través de métodos de Deep Learning, utilizando tanto fotopletismografía remota como de contacto. Consideramos que este método será de sumo valor para las teleconsultas y el automonitoreo de los pacientes, especialmente al obtener una fotopletismografía. La predicción de dicha señal facilita la estimación de múltiples parámetros fisiológicos para una mayor evaluación del estado del paciente, por ejemplo: variabilidad de la frecuencia cardíaca, saturación de oxígeno en sangre y estrés. Sin embargo, dada la variabilidad observada en diferentes escenarios, creemos que son necesarias futuras investigaciones para ofrecer un método robusto y confiable que pueda ser utilizado en la práctica clínica.

Capítulo 1

Introducción y motivaciones

De acuerdo a la Declaración Universal de Derechos Humanos de 1948, '*toda persona tiene derecho a un nivel de vida adecuado que le asegure, así como a su familia, la salud y en especial la alimentación, el vestido, la vivienda, la asistencia médica y los servicios sociales necesarios*' [15]. La salud es un derecho humano fundamental, y lograr su cobertura universal es uno de los principales objetivos de las naciones, buscando asegurar el más alto nivel realizable de servicios para todas las personas.

La cobertura sanitaria universal implica que todas las personas y comunidades reciban los servicios de salud cuando y donde lo necesiten, de una buena calidad, y sin tener que pasar penurias financieras para pagarlos. Abarca toda el espectro de servicios de salud esenciales, desde la promoción de la salud hasta la prevención, el tratamiento, la rehabilitación y los cuidados paliativos [16].

Según el reporte global de cobertura universal de la salud y protección financiera en salud de la Organización de las Naciones Unidas (ONU) en conjunto con el Banco Mundial en 2021 [1], la cobertura de servicios mejoró lentamente desde un índice promedio global de 45 en el 2000 a 67 en 2019. No obstante, la proporción de la población que gastan al menos un 10 % de su presupuesto familiar para pagar los servicios de salud creció desde un 9.4 % en el 2000 a 13.2 % en 2017. Entre ellos, 290 millones de personas gastaron más del 25 % de sus presupuestos en salud y se estima que aproximadamente 70 millones de personas fueron empujadas a la pobreza extrema en 2017 por los gastos de bolsillo en salud, mientras que otros 435 millones la gente fue empujada más profundamente a la pobreza extrema por gastos en salud.

La pandemia del COVID-19 ha tenido un impacto significativo en la salud y economía. Entre los problemas desencadenados se encuentran el aumento de la inequidad; la dificultad de pagar los servicios, particularmente en poblaciones de bajo ingreso económico; la disrupción de la prestación de servicios esenciales de salud y la ralentización de la eficacia de la respuesta del sistema de salud. En promedio, los países informaron que alrededor del 56 % de los 28 de los servicios esenciales a realizar se-

guimiento se habían interrumpido en el tercer trimestre de 2020 y el 41 % seguían interrumpidos a principios de 2021. A finales de 2021, el alcance de las interrupciones del servicio informado por los países se mantuvo similar a los niveles de principios de 2021 (44 %). Todos los entornos de atención de la salud y las plataformas de prestación de servicios se vieron afectados, en particular los servicios de atención primaria.

A esto se le suma las limitaciones de un sistema de salud sin suficientes trabajadores. En 2016, la Organización Mundial de la Salud (OMS) proyectó un déficit mundial de 18 millones de trabajadores de la salud para 2030, particularmente en las regiones de África y el Sudeste Asiático de la OMS (ver figura 1.1). Y, según la segunda ronda de la encuesta global sobre la continuidad de los servicios esenciales de salud [1], dos tercios (66 %) de los países informaron que la escasez de personal fue una razón importante para las interrupciones del servicio de enero a marzo de 2021.

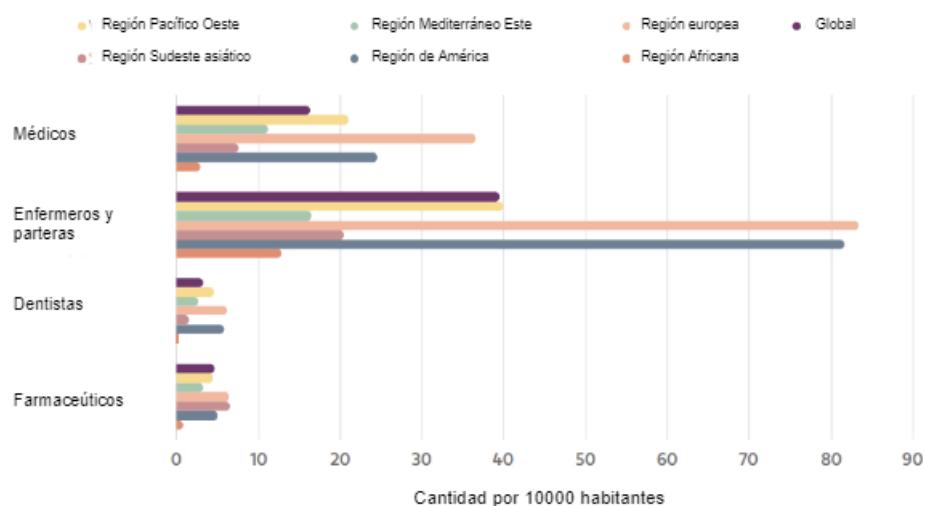


Figura 1.1: Densidad de médicos, enfermeros, parteras, dentistas, farmacéuticos por 10000 habitantes, según la región de la ONU, en 2022 [1].

Adicionalmente, la cantidad de personas mayores de 60 años en 2019 ascendió a los 1.000 millones, y se espera que aumente a 2100 millones para 2050[17]. Esto, en conjunto con el crecimiento de la población, implica que hay un aumento de personas que precisan una atención en salud personalizada, al día, sin espera o retrasos, de la mejor calidad posible. Sin embargo, hoy por hoy, hay un desajuste de 18 millones de profesionales entre el personal sanitario en activo en todo el mundo actualmente y el que se necesitaría hoy en día para garantizar la cobertura sanitaria universal [18, 19]. La demanda al sistema de salud puede sobrecargar los hospitales y sistemas, especialmente en el contexto de una pandemia como la provocada de COVID-19.

A la situación planteada se le suman otros aspectos que dificultan brindar la mejor asistencia posible, tales como la falta de información o la presencia de información confusa. La información nace del procesamiento de grandes cantidades de datos los

cuales son interpretados para poder realizar acciones, y en este caso en específico, ayudan a diagnosticar a los pacientes y mejorar su calidad de vida. Una falta de datos o una gran cantidad de datos sin distinguir aquellos verdaderamente importantes puede dificultar el entendimiento de la condición de los pacientes.

En este contexto, la teleconsulta mediante dispositivos móviles ha demostrado ser útil en el diagnóstico y monitoreo de los pacientes. Sin embargo, la incapacidad de realizar un examen físico completo limita sus beneficios y adopción. Por lo tanto, resulta de vital importancia el desarrollo de tecnologías y herramientas que puedan brindar información, sin la necesidad de grandes inversiones. El desarrollo de tecnología especializada en el bienestar y salud, basados en software e inteligencia artificial, ha demostrado ser capaz de brindar una solución a esta problemática [20–23]. En particular, aquellas soluciones capaces de ser utilizadas desde dispositivos remotos comerciales y sin contacto serían herramientas a las que los mismos pacientes podrían acceder para llevar un control periódico de su propia situación y contactar un profesional al requerirlo.

En Argentina, de acuerdo al informe técnico de Ciencia y Tecnología del INDEC para el cuarto trimestre de 2021 [2], los datos muestran que 88 de cada 100 argentinos usan teléfono celular y 87 de cada 100 utilizan internet (ver figura 1.2).

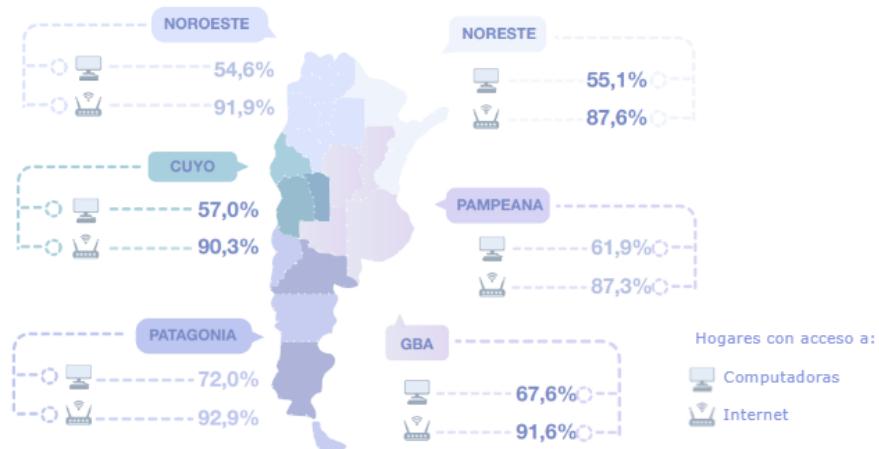


Figura 1.2: Porcentaje de hogares con acceso a computadora e internet, según región argentina en el cuarto trimestre de 2021[2].

En el segundo trimestre de 2022, según el informe técnico de Servicios del INDEC[3], se registró un aumento de 1,3% respecto al segundo trimestre de 2021 en acceso a internet fijo. En cuanto a los accesos a internet móviles, en el trimestre bajo análisis, se observó una suba del 7,8% respecto al mismo trimestre del año anterior (ver figura 1.3).

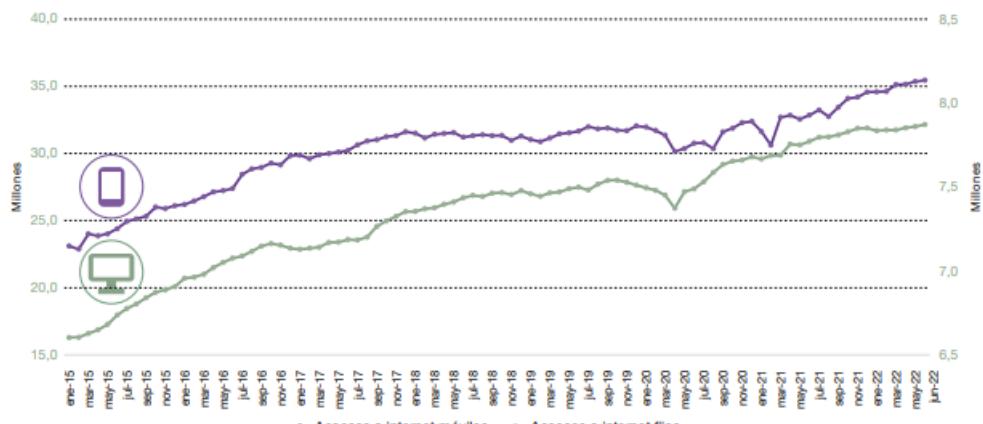


Figura 1.3: Accesos a internet fijos y móviles. Enero 2015- junio 2022, Argentina[3].

Debido a que una gran parte de la población argentina tiene acceso a computadoras e internet, la extracción de parámetros fisiológicos a través de software y dispositivos móviles (computadoras, teléfonos celulares, etc) podría proveer alternativas a la necesidad de dispositivos o sensores especiales de alto costo. En conjunto con la telemedicina, podría generar una extensión de los servicios de atención en salud de forma digital en cualquier lugar, dejando al alcance de más personas la posibilidad de un mejor servicio; y mayor accesibilidad a aquellos datos que no se podrían monitorizar por el personal de salud debido a circunstancias especiales, como lo es una pandemia y sus protocolos de distanciamiento preventivo o cuarentena, entre otros.

1.1. ÜMA

Este trabajo fue realizado en el contexto de práctica laboral en la empresa Üma Health S.A.

Üma Health es una start-up internacional del segmento Health Tech. Busca combinar conocimiento médico con tecnología basadas en inteligencia artificial (IA) para mejorar la accesibilidad y calidad a los servicios de salud y bienestar a través de su plataforma, ayudando a los médicos a brindar un servicio más efectivo.

La empresa lleva modelos de negocio tanto de B2B (del inglés: *Business to Business*) como B2C (del inglés: *Business to Costumer*), es decir, que la empresa provee servicios de salud tanto a empresas como a consumidores finales.

Diagnósticos inteligentes, clasificador de antecedentes y predicción de enfermedades son algunas de las herramientas que se desarrollan en ÜMA para hacer más eficiente la experiencia del usuario en la plataforma complementando su servicio de consultas online.

El equipo de alto rendimiento, internacional y multidisciplinario, está formado por 40 profesionales, entre ellos ingenieros de software, científicos, psicólogos y médicos.

Desde su lanzamiento en 2020, ÜMA Health se encuentra activa en la Argentina y, recientemente, México. Su sede principal se encuentra en Coghlan, Ciudad Autónoma de Buenos Aires, Argentina.

1.2. Estado del arte

Para llevar a cabo un examen médico, hay diversos equipos médicos especializados dependiendo del parámetro fisiológico que se requiera. La frecuencia cardíaca es obtenida típicamente a partir de la técnica de fotopletismografía (FPG). La fotopletismografía es utilizada para medir la variación del volumen de sangre a partir de la reflexión de un haz de luz sobre la piel.

La técnica de FPG es utilizada por varios dispositivos comerciales tales como muñequeras o relojes smart, que registran las señales biológicas, o saturómetros, cuyo precio en algunos casos lo hace inaccesible. Estos dispositivos precisan de contacto, por lo cual no pueden ser utilizados para monitorear recién nacidos o pacientes con piel frágil. Además, monitoreo bajo un período largo de tiempo puede llevar a incomodidades e incluso riesgo de infecciones en la piel. Una solución a estos problemas es conocida como FPGs remotos los cuales han ganado tracción en la última década. No obstante, este método requiere un procesamiento más profundo para obtener la mejor señal posible sin contacto [24].

En investigaciones iniciales, se utilizaron técnicas de procesamiento de señales más tradicionales. Entre estas, podemos mencionar: el análisis de componentes independientes (ICA) [25] o el análisis de componentes principales (PCA) [26]. Otros métodos surgieron para superar las problemáticas que enfrentaban respecto a los ruidos por movimiento tales como CHROM [27], 2SR [28] o POS [29]. Aunque estos algoritmos demostraron buenos resultados en videos de la cara, la mayoría lo hacía bajo condiciones controladas de iluminación, movimiento, o falta de variedad en los tonos de piel.

Con el crecimiento de la investigación en los campos de inteligencia artificial, especialmente en modelos de *Computer vision*, algoritmos de diversos tipos basados en inteligencia artificial empezaron a aplicarse en la obtención de señales biológicas. Algunos algoritmos de IA abarcan todo el proceso de procesamiento hasta conseguir el valor final de predicción (*end-to-end*).

Los primeros métodos *end-to-end* de extracción de FPG por videos faciales a mencionar son aquellos basados en redes convolucionales (CNN) 2D tales como DeepPhys [30] y HR-CNN [31]. Sin embargo, estos métodos solo toman información espacial de los *frames* del video, por lo tanto, se han propuestos diferentes tipos de frameworks de redes 3D para también usar información temporal de las señales fisiológicas en los videos, por ejemplo, CNN PhysNet [32], rPPGNet [33], AutoHR [34], entre otros [35–37]. Tam-

bién se han desarrollado más arquitecturas que extraigan información espacio-temporal evitando los procesos de cómputo complejo de las redes 3D: una combinación de redes 2D con redes neuronales recurrentes, ya sean LSTM, BiLSTM o ConvLSTM [32].

Por otro lado, existen lo que se conocen como métodos híbridos de Deep Learning para obtención de las mediciones. Estos métodos solo aplican Deep Learning a cierta parte del flujo del algoritmo. Algunos de estos algoritmos optimizan la obtención de la señal al focalizarse en la segmentación de la piel para obtener la señal más fuerte y con menos artefactos [38, 39]. Otros algoritmos híbridos se centran en la extracción de la señal del video. Dentro de los algoritmos híbridos se han propuesto múltiples desarrollos de Deep Learning para obtener una señal de fotopletismografía de alta calidad: arquitecturas de LSTM [40, 41], CNNs 2D [38, 42], CNNs 3D [43–45] e incluso, CNNs 3D con redes neuronales recurrentes [46]. Por último, los algoritmos híbridos de estimación de la frecuencia cardíaca a partir de la señal de fotopletismografía. La estimación de la frecuencia cardíaca puede ser realizado por un filtro pasa-bandas seguido por un análisis de frecuencia o detección de picos en la forma de procesamiento tradicional. Sin embargo, también puede ser clasificado como un problema de regresión y resuelto por métodos de Deep Learning variados, por ejemplo:

- CNNs en una dimensión sobre la señal de FPG [47] para la predicción de la frecuencia cardíaca.
- CNNs en dos dimensiones para analizar el espectrograma de la señal de FPG como imagen [48].
- CNNs en dos dimensiones para analizar mapas espacio-temporales de la FPG [49, 50].

La fotopletismografía remota no solo puede extraerse desde grabaciones de la cara si se busca obtenerse a través de métodos de contacto con teléfonos celulares. Extraer dicha señal a través de grabaciones del dedo con cámaras, análogas a lo que sería un oxímetro tradicional, se han desarrollado también. Jonathan and Leahy [51] en 2010 demostró en un caso de estudio que la FPG generada a partir del dedo por la cámara de un celular de grado consumidor (Nokia model E63) contenía los cambios de intensidad para detectar la frecuencia cardíaca. Siddiqui et al. [52] presentó un método de extracción de FPG utilizando las intensidades de los píxeles que superaban cierto umbral cargado a mano. Al ser elegido el umbral por el mismo usuario dado que varía dependiendo de la persona, se reduce la versatilidad del sistema. En Yan et. al. [53] se dispusieron a analizar la exactitud de un algoritmo (Cardiio) de estimación de la frecuencia cardíaca para el dedo en estado de reposo, ejercicio moderado y ejercicio intenso.

Dada la variedad de investigaciones y desarrollos, en la tabla 1.1 se muestran los métodos considerados más relevantes para la comparación del trabajo. Las métricas con las que se comparan a los modelos son el desvío estándar (SD), el error absoluto medio (MAE), el error cuadrático medio (RMSE) y la correlación de Pearson (ρ).

Método	SD	MAE	RMSE	ρ
Poh et. al.[25]	24.3	25	25.9	0.08
CHROM[27]	-	13.49	22.36	0.21
HR-CNN[31]	-	7.25	9.24	0.51
DeepPhys[30]	-	4.57	-	-
PhysNet[32]	7.8	5.96	7.88	0.76
Auto-HR[34]	4.73	3.78	5.1	-
Rencheng[50]	5.57	4.61	5.70	0.86
Siddiqui et al. [52]	-	3	-	-
Yan et. al. [53]	-	-	1.03 a 2.15	-

Tabla 1.1: Desempeño de la estimación de la frecuencia cardíaca de los desarrollos relevantes a ser comparados con el modelo final con dataset públicos [14].

A lo largo de este trabajo, se han intentado aplicar técnicas de Deep Learning basada en algunos de los métodos mencionados anteriormente, para determinar frecuencia cardíaca a partir de señales de video de la cara y el dedo.

1.3. Objetivos

El objetivo planteado para este trabajo fue el desarrollo de un algoritmo robusto de Machine Learning que permita la estimación de la frecuencia cardíaca mediante fotopletismografías utilizando cámaras de los dispositivos de uso personal tales como smartphones y computadoras.

Para este objetivo, se define la 'robustez' de un algoritmo como el grado en el cual el rendimiento de un modelo se modifica al utilizar nuevos datos de entrada, en comparación con los datos de entrenamiento. Idealmente, el rendimiento no debería desviarse significativamente frente a nuevos datos, especialmente si el modelo tiene la finalidad de ser utilizado en un futuro dentro de un producto comercial frente a una variedad de dispositivos y condiciones.

Capítulo 2

Marco teórico

2.1. Fotopletismografía

La fotopletismografía es una técnica óptica de bajo costo y no invasiva para el monitoreo de parámetros tales como la frecuencia cardíaca, la saturación de oxígeno en sangre, variabilidad de la frecuencia, entre otros. Esta utiliza las variaciones del volumen de sangre por debajo de la piel que son producto de la naturaleza pulsátil del sistema circulatorio. Es sobre esta técnica que no solo se comparó el algoritmo desarrollado, sino que es a través de la cual se sustenta.

La fotopletismografía consiste en iluminar un tejido, y detectar la intensidad de luz reflejada (o transmitida) a través del mismo (ver figura 2.1). Los cambios de perfusión por el volumen de sangre que pasa a través de los vasos sanguíneos producen variaciones en la intensidad de la luz detectada y permiten generar una señal pulsátil a partir de la cual se obtiene la información buscada [54].

El principio fundamental del FPG se basa es el de la ley de Lambert-Beer, en conjunto con el distinto comportamiento que presentan la sangre y otros componentes del tejido al ser iluminados por diferentes longitudes de onda. La ley de Lambert-Beer establece que la absorción de la luz, A , por la sangre es definida de la siguiente forma:

$$A = \log \left(\frac{I_0}{I} \right) = \alpha \cdot L = E \cdot C \cdot L, \quad (2.1)$$

donde A es la absorbancia, I_0 es la intensidad de la luz incidente, I es la intensidad de la luz reflejada (o transmitida), α es el coeficiente de absorción, E es el coeficiente de extinción ($dL/(g \cdot cm)$), C es la concentración (g/dL) y L es la longitud atravesada por la luz en el medio. Se debe considerar que el coeficiente de absorción depende del material y la longitud de onda modificando así la intensidad de la luz reflejada (o transmitida) en relación a la intensidad incidente.

Además, dado que la interacción de la luz con el tejido biológico puede presentar un comportamiento complejo (en cuanto a dispersión, absorción, reflexión y otros fenóme-

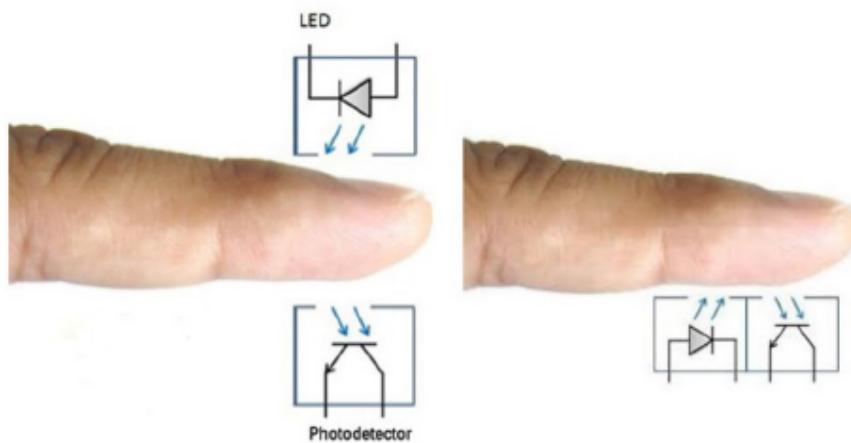


Figura 2.1: Arreglo de los sensores y LED en fotopletismografías. A la derecha, por reflexión. A la izquierda, por transmisión [4].

nos lumínicos), la elección de la longitud de onda es importante. La longitud de onda elegida debe tener una gran absorción para con la sangre comparada a otros componentes del tejido, debido a que esto permitiría una detección más exacta de los cambios del volumen de sangre dentro del tejido. Las longitudes de onda más cortas de la luz son fuertemente absorbidas por la melanina, mientras que el agua absorbe luz en el rango de la luz ultravioleta e infrarroja. Por lo tanto, luz roja (660 nm) y cercana al rango infrarrojo (940 nm) son las fuentes de luz más comúnmente utilizadas en sensores de fotopletismografía [4].

Por otro lado, la absorción de la luz verde (500–600 nm) por la oxihemoglobina (O_2Hb) y deoxihemoglobina (RHb) es mucho mayor que la luz infrarroja lo que produce una mayor diferencia entre el ciclo cardíaco sistólico y diastólico y una mejor relación señal-ruido. Esto implica que puede alcanzarse mayor exactitud a la hora de detectar la frecuencia cardíaca que al procesar las señales obtenidas por el infrarrojo [55][56][57].

Cuando el corazón bombea la sangre a través del cuerpo durante la sístole, la cantidad de sangre que llega a los capilares en la superficie de la piel aumenta, resultando en mayor absorción de la luz. La sangre viaja hacia el corazón por el circuito venoso, llevando a una disminución del volumen y, por ende, de la absorción de la luz. La onda de fotopletismografía medida resultante (ver figura 2.2) es una onda fisiológica pulsátil (usualmente denominada 'AC') que refleja los cambios sincrónicos cardíacos del volumen sanguíneo con cada latido. Esta onda se encuentra superpuesta a una señal constante base mucho más lenta con variaciones quasi-estáticas ('DC'). Esta componente 'DC' contiene información valiosa acerca de la respiración, el flujo venoso, las actividades del sistema nervioso simpático y la actividad termoregulante [54].

Hay básicamente dos configuraciones geométricas posibles de las posiciones de la fuente de luz y del detector: obtención por reflexión y obtención por transmisión (ver figura 2.1). Específicamente, en el modo de transmisión, una muestra del tejido es

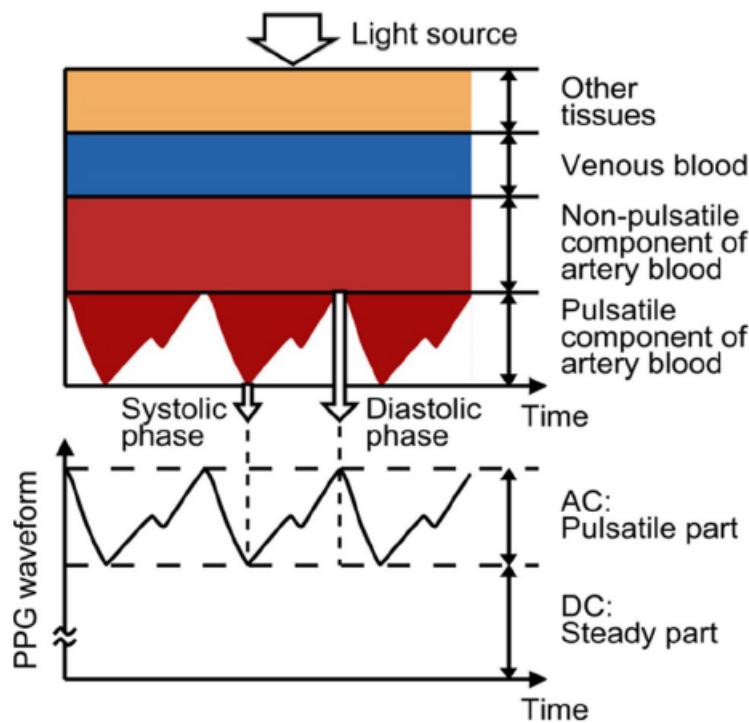


Figura 2.2: Componentes de la señal de fotopletismografía [4].

puesta entre la luz y el fotodetector, mientras en el modo de reflexión, la fuente y el detector son colocados del mismo lado uno al lado del otro.

En el modo de transmisión, el fotodetector debe sensar la luz transmitida a través del tejido y, por lo tanto, el sitio de medición está limitado a tejidos donde la luz transmitida pueda ser detectada rápidamente, por ejemplo, puntas de los dedos o el cartílago de la oreja irrigado. No obstante, estos sitios de medición son susceptibles a ser condicionados por ambientes extremos como baja temperatura ambiental o, el sensor puede interferir con actividades diarias.

Conceptualmente, en el modo reflectivo, el fotodetector mide la luz que es dispersada hacia atrás o reflejada en el tejido, hueso y/o sangre. Esta configuración permite las mediciones desde cualquier punto en la superficie de la piel, incluyendo aquellas que son de difícil acceso para el modo reflectivo. De todos modos, de ser un sensor óptico, este debería estar firmemente en contacto con la piel, por lo cual requiere un sitio con un área de piel plana.

A pesar del amplio rango de aplicaciones de la fotopletismografía, hay varias desventajas que pueden limitar su usabilidad y evolución en su forma de sensor óptico convencional:

- El sensor solo puede monitorear el cambio del volumen de sangre en un sitio o punto de muestra. Si bien mediciones con múltiples sensores han sido desarrolladas, estos requieren sincronización electrónica y óptica de cada sensor y simetría

de la posición en la que se encuentran (ambos lóbulos de la oreja, ambas piernas, etc) e impiden movimiento del paciente, dan incomodidad al paciente y no es apropiado para uso rutinario.

- Para mediciones exactas, el sensor de fotopletismografía convencional precisa tener contacto directo firme, que limita su uso en caso de diagnósticos de heridas como quemaduras, úlceras; evaluación de piel o que se requiera movimiento.
- La fotopletismografía es susceptible a artefactos y ruido causado por movimiento. Dentro de la medición en un solo punto, la remoción de estos artefactos o su atenuación presenta un gran desafío en el procesamiento de la señal.

2.2. Fotopletismografía remota

La fotopletismografía remota (rFPG) o de imagen es una técnica reciente que incluye el uso de una cámara de video sin contacto directo. El principio del monitoreo por rFPG es similar al del modo reflectivo tradicional. En vez del fotodiodo se utiliza una matriz-imagen de dos dimensiones. En el uso de la cámara RGB (rojo-verde-azul), la luz verde es la que provee la señal de FPG más fuerte. La hemoglobina absorbe mejor la luz verde que la roja y penetra en la piel de forma más profunda que la azul. Es por esto que la luz verde extrae una señal pulsátil con una marcada diferencia entre la fase sistólica y la diastólica y la señal es usualmente la más utilizada para calcular la frecuencia cardíaca.

En comparación con la técnica tradicional, rFPG utiliza luz ambiente y no requiere una fuente de luz específica. Aún más, el sensor de la cámara permite la extracción de información separada espacialmente, que supera la limitación de mediciones en lugares específicos de la piel.

La rFPG puede ser realizada sin contacto de una larga distancia usando un dispositivo de carga acoplado a la cámara para medir la luz reflejada en la piel. Alternativamente, se pueden realizar rFPG con contacto, que habilita mediciones a distancias cortas con una cámara de celular.

Las señales de pletismografía pueden ser medidas usando luz ambiente que contiene múltiples longitudes de onda (660, 810, y 940 nm) y un detector de cámara digital o celular. Las señales de frecuencia cardíaca y oximetría son determinadas por el procesamiento de la señal de los canales RGB. Aunque el canal verde contenga la señal más fuerte de fotopletismografía, los otros dos canales contienen también información.

2.3. Inteligencia Artificial y salud

Inteligencia Artificial, según IBM, es un campo que combina la ciencia informática y los conjuntos de datos robustos para permitir la resolución de problemas [58]. En los años recientes ha surgido tanto interés como avances en las aplicaciones médicas de la inteligencia artificial dado el poder de cómputo de las modernas computadoras y la gran cantidad de datos digitales disponibles para su uso.

Hay muchas aplicaciones de IA en la medicina que pueden ser utilizadas en una variedad de campos médicos tales como las prácticas clínicas, diagnósticas, rehabilitativas, quirúrgicas y predictivas [59]. Las tecnologías basadas en IA pueden consumir, analizar y reportar grandes volúmenes de datos para detectar patologías y guiar a los médicos en sus decisiones. Dichas aplicaciones además pueden lidiar con una vasta cantidad de datos producidos en el contexto médico y hallar nueva información que de otra forma permanecería oculta en la masiva cantidad de datos médicos. Con múltiples aplicaciones, la Inteligencia Artificial asiste médicos y profesionales de la salud en los dominios de los sistemas informáticos de salud, datos de salud geocodificada, vigilancia sindrómica y epidemiológica, o modelos predictivos o diagnóstico de enfermedades, sistemas de soporte de decisiones e imágenes médicas.

Un aspecto notable de las técnicas de Inteligencia Artificial aplicadas a la Salud es el apoyo en la administración de los servicios de salud. Estas aplicaciones pueden dar soporte a doctores, enfermeras y administradores en el trabajo. Por ejemplo, los sistemas IA pueden proveer a los profesionales actualizaciones de la información médica de forma constante y a tiempo real de varias fuentes, incluyendo diarios, textos académicos, y prácticas clínicas, especialmente en situaciones críticas de filtrado de información como lo puede ser una pandemia. Otras aplicaciones pueden incluir coordinar herramientas de información para pacientes y dar acceso a inferencias apropiadas para alertas de riesgos en la salud. Es decir, que la inteligencia artificial permita a los hospitales y servicios de salud trabajar de forma más eficiente porque:

- Les da acceso a datos clínicos inmediatamente cuando se los necesita.
- Los profesionales pueden actuar conforme a los datos procesados y bajo las recomendaciones de sistemas de soporte.
- Los pacientes pueden mantener informados y participar de su propio cuidado al comunicarse con los equipos médicos durante sus tratamientos.

Otro aspecto relevante de IA en el ámbito de la salud es la predicción de enfermedades, diagnósticos y tratamiento, porque IA puede identificar relaciones significativas en los datos crudos, puede ayudar en el diagnóstico, tratamiento y predicción de resultados en diversas situaciones médicas. Esta tecnología permite a los profesionales tomar

un rol proactivo en el tratamiento de una enfermedad identificando posibles factores de riesgo, personalizando el tratamiento para el mejor resultado posible. Así también, los doctores se benefician al tener más tiempo y datos concisos para generar mejores decisiones informadas. Esto es acompañado por un procesamiento de una gran cantidad de datos médicos disponibles dado que la Inteligencia Artificial puede manejar datos provenientes de actividades clínicas (tratamientos, diagnósticos, imágenes) y asociaciones del personal con la temática de interés. Un ejemplo de lo mencionado es el aumento de diagnósticos remotos a través de la telemedicina que permite una observación del paciente con herramientas de soporte para los profesionales.

2.4. Machine Learning y Deep Learning

Dentro de la Inteligencia Artificial [58], podemos encontrar la rama de *Machine Learning* o Aprendizaje Automático, que está compuesta por algoritmos de IA que buscan crear sistemas expertos que hagan predicciones o clasificaciones basándose en datos de entrada. *Machine Learning* se focaliza en el uso de los datos y algoritmos para imitar la forma en la que el humano aprende, gradualmente mejorando su precisión. A su vez, el *Deep Learning* o Aprendizaje Profundo es un subcampo de *Machine Learning* el cual se compone de redes neuronales.

Mediante métodos estadísticos, los algoritmos son entrenados para realizar clasificaciones o predicciones, y descubrir perspectivas claves en proyectos con gran cantidad de datos. Conforme el algoritmo ingiere datos de entrenamiento, es posible producir modelos más precisos basados en esos datos. Su aprendizaje es iterativo lo cual conduce a una mejora en los tipos de asociaciones hechas o patrones entre los elementos de datos. Una vez entrenado el modelo, cuando se le proporcione datos nuevos, el modelo retornará un pronóstico basado en los datos que entrenaron al modelo.

El aprendizaje de estos algoritmos puede ser supervisado o no supervisado. Se define como supervisado por el uso de conjuntos de datos etiquetados utilizados para entrenar algoritmos que clasifiquen datos o realizar predicciones de forma precisa [60, 61]. A medida que los datos de entrada alimentan el modelo, el modelo modifica los pesos hasta que ajusta el modelo apropiadamente. El aprendizaje supervisado ayuda a resolver una variedad de problemáticas aplicables en el mundo real en escala. Algunos de los métodos utilizados en el lenguaje supervisado incluyen redes neuronales, Naive Bayes, regresión neuronal, regresión logística, Random Forest y máquina de vectores soporte [61]. En cambio, el aprendizaje no supervisado usa algoritmos de Machine Learning para analizar y *clusterizar* conjuntos de datos sin etiquetar [60, 62]. Estos algoritmos descubren patrones ocultos en los datos por si mismos sin necesidad de la intervención humana en el entrenamiento, por esto se define como no supervisado. Sin embargo, siguen requiriendo de intervención para validar los resultados.

Particularmente, los algoritmos utilizados en este trabajo se encuentran bajo la clasificación de Machine Learning supervisado. En este trabajo, los datos de entrada son los videos y se encuentran asociados a una etiqueta la cual puede ser un valor escalar de frecuencia cardíaca o una señal de fotopletismografía dependiendo del modelo. Estas etiquetas son el objetivo a predecir con el entrenamiento de diversas redes neuronales, algoritmos de Deep Learning.

El Deep Learning es un método específico de Machine Learning que incorpora las redes neuronales en capas sucesivas para aprender de los datos de manera iterativa [58]. El Deep Learning es especialmente útil cuando se trata de aprender patrones de datos no estructurados. Las redes neuronales complejas de Deep Learning están diseñadas para emular cómo funciona el cerebro humano, por lo que las computadoras pueden ser entrenadas para lidiar con abstracciones y problemas mal definidos. Los algoritmos de Deep Learning generalmente están creados utilizando frameworks que aceleran el desarrollo de la solución, como Tensorflow o Pytorch.

Lograr una predicción usando métodos de Machine Learning convencional requiere numerosos pasos secuenciales, específicamente de preprocessamiento, extracción de características, selección de características, entrenamiento y clasificación. La extracción y selección de características es un aspecto fundamental debido a su impacto en el desempeño de las técnicas de Machine Learning. Una selección de características sesgada puede generar una mala discriminación entre clases. En contraposición a los métodos convencionales de Machine Learning, los algoritmos de Deep Learning tienen la habilidad de automatizar el aprendizaje de los conjuntos de características para muchas tareas, permitiendo que el aprendizaje y la clasificación pueda ser alcanzada en un mismo tiempo [9].

Dentro de las características del Deep Learning, podemos encontrar:

1. La habilidad de ser aplicado en aproximadamente todos los dominios de aplicación, o mejor conocido como aprendizaje universal.
2. No requiere características diseñadas para una solución en específico en Deep Learning. En cambio, las características optimizadas son aprendidas de forma automatizada en relación a la tarea y objetivo a lograr. De esta forma, el Deep Learning adquiere robustez frente a los cambios de datos de entrada.
3. Una misma técnica de Deep Learning puede ser ajustada a diferentes aplicaciones o tipos de datos a través del Transfer Learning [63].
4. Es altamente escalable y con una gran capacidad de manejar fuentes de datos con gran caudal, conocido como Big Data.

Con lo anteriormente mencionado en mente, en las siguientes secciones, se profundizan los aspectos claves respecto al Deep Learning para comprender el desarrollo del trabajo.

2.5. Redes Neuronales Artificiales

El reconocimiento de patrones [64] es un paradigma computacional utilizado en la clasificación de los datos crudos. El mismo adopta una variedad de enfoques que proveen el desarrollo de múltiples aplicaciones en varios campos de actividad. El objetivo de estos enfoques es imitar la inteligencia humana.

Un patrón puede ser referido como un conjunto de elementos, objetos, imágenes, eventos, casos, situaciones, características o abstracciones donde las facetas del conjunto son iguales en un sentido inequívoco. También puede ser definido por el denominador único o recurrente a través de múltiples muestras de la entidad, por ejemplo, el común denominador en imágenes de huellas dactilares puede definir el patrón de una huella dactilar. Entonces, un patrón puede ser la imagen de una huella dactilar, una cara humana, palabras escritas a mano, un código de barras, una página de internet o una señal de audio; mientras que el reconocimiento es la tarea de identificar en ellos un objeto, un evento o una característica.

Las técnicas convencionales aplicadas para manejar problemas de reconocimiento de patrones son clasificadas en enfoques estructurales, estadísticos o híbridos. No obstante, los enfoques estructurales y estadísticos por separado pueden producir resultados no satisfactorios si son aplicados como soluciones separadas a un problema complejo de reconocimiento de patrones. Por ejemplo, al ser aplicado, el método estructural puede ser débil a la hora de manejar patrones ruidosos. Por otro lado, el método estadístico es incapaz de utilizar información correspondientes a patrones de estructuras. Es por esto que se decidió utilizar el método híbrido para combinar ambos tipos de métodos. De todos modos, los modelos de redes neuronales artificiales son utilizados hoy en día ya que presentan mejores resultados de forma notoria en reconocimiento de patrones sobre tareas con múltiples complejidades.

Las Redes Neuronales Artificiales (ANNs, por sus siglas en inglés *Artificial Neural Networks*) [5, 65, 66] son modelos matemáticos estadísticos no lineales que han sido creados a partir del funcionamiento del cerebro humano. Sin embargo, los modelos no tienen el objetivo de generar un modelo biológicamente realista, sino que su objetivo es el de analizar datos.

El propósito de la IA es crear una máquina que puedan trabajar como el cerebro humano y sean capaces de 'pensar'. Las redes neuronales tratan de imitar dicho funcionamiento con elementos llamados nodos (que emulan a las neuronas) y las uniones entre ellas con funciones de transferencia. De esta forma, han incrementado su atractivo, efectividad, eficiencia y éxito a la hora de lograr el reconocimiento de patrones en diversas problemáticas. Como el cerebro, las ANNs van a distinguir los patrones, manejar hechos y cifras, y ser entrenadas.

Una neurona biológica o artificial adquiere una cierta cantidad de entradas ya sea de

información presente o de las salidas de neuronas relacionadas de capas anteriores. Cada entrada se relaciona a través de una asociación, llamada sinapsis, que se caracteriza por un peso. Una célula nerviosa, del mismo modo, presenta un valor umbral, de forma que si la sumatoria de los pesos supera ese valor, la neurona se excita y la indicación de estimulación construye la salida de la célula nerviosa. Esto se imita en las redes neuronales artificiales.

Típicamente en ANNs tradicionales están compuestas por capas que, a su vez, están compuestas por un arreglo sistemático de 'neuronas' o nodos tal como las redes neuronales biológicas. Una red neuronal artificial puede ser representada como una interpretación aritmética lineal del diseño de una neurona individual, reflejando su capacidad de 'aprendizaje' y 'generalización'.

Además, las neuronas de una capa están relacionadas con las neuronas de la capa siguiente y estas relaciones tienen pesos. Estas capas pueden ser diferentes entre sí dependiendo de sus características y función dentro de la red neuronal. Por ejemplo, la capa de entrada o input es por la cual los datos conocidos son introducidos al modelo. Luego están las capas intermedias conocidas como capas ocultas. Por último, la capa de salida o output, de la cual el valor final a predecir es obtenido. Cada capa consiste en estos nodos/neuronas y cada una está conectada a la capa siguiente a través de funciones de transferencia.

En una ANN (ver figura 2.3), la salida o output de la capa $i-1$ es la entrada o input de la capa i . Los datos conocidos son introducidos en la capa de entrada aplicándole operaciones matemáticas que se desarrollan en más profundidad en las siguientes secciones. El resultado final de cada capa es transferido por funciones de transferencia a la próxima capa hasta que se alcanza la capa final, cuyo valor es tomado como el resultados de la red neuronal.

2.5.1. Perceptrón simple

El perceptrón de una sola capa es la unidad básica de la red neuronal artificial. Un perceptrón consiste en valores de entrada, pesos, un valor de bias, una suma ponderada y una función de activación.

El perceptrón trabaja de forma que toma valores numéricos como entrada junto a los pesos y los bias. Se realiza la suma ponderada multiplicando las entradas con los respectivos pesos para que luego los productos resultantes sean sumados entre si y con los bias. Finalmente, la función de activación toma la suma ponderada y retorna la salida final (ver figura 2.4). El output 'Y' de la neurona está dado por

$$Y = bias + \sum_{i=1}^5 x_i \cdot w_i. \quad (2.2)$$

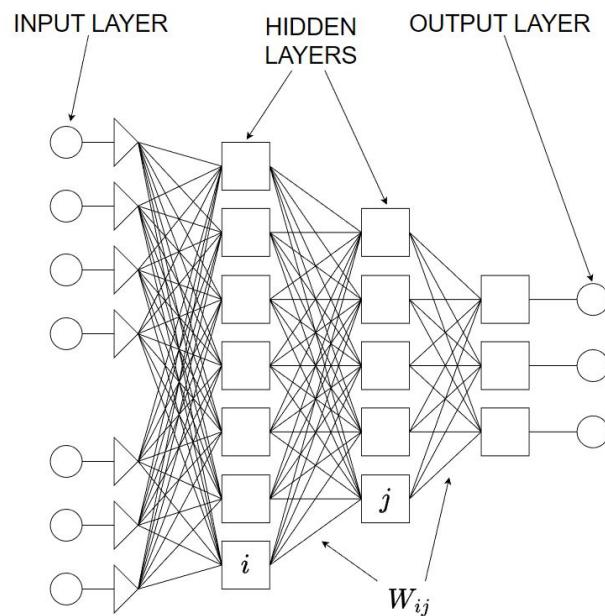


Figura 2.3: Diagrama de una red neuronal artificial convencional con 3 capas ocultas, una capa de entrada y una capa de salida. Los pesos que caracterizan las uniones entre la capa j y la capa anterior i son denominados w_{ij} [5].

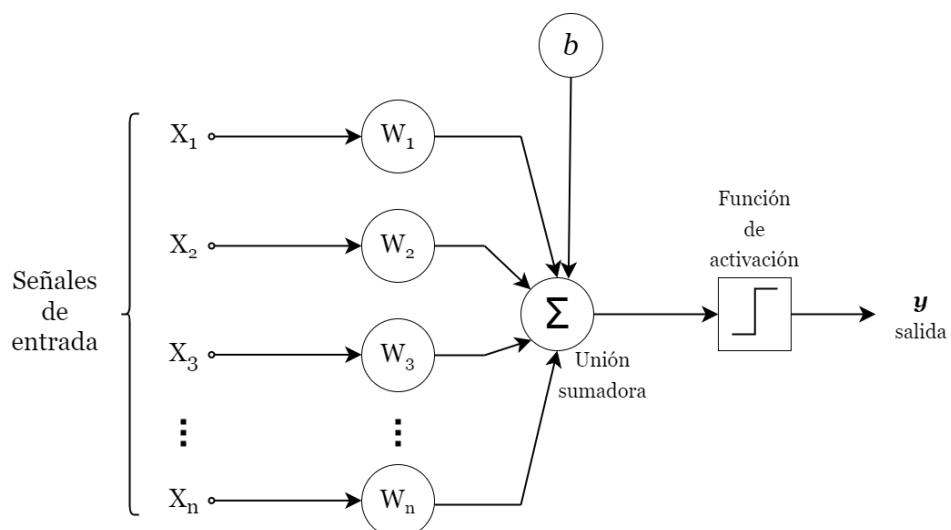


Figura 2.4: Diagrama de un perceptrón simple y sus componentes. Hay una única neurona con n inputs $x_1, x_2, x_3, x_4, \dots, x_n$. Estas entradas son multiplicadas por sus pesos respectivos ($w_1, w_2, w_3, w_4, \dots, w_n$) y sumado el bias b correspondiente.

Que dicho output active o no la siguiente neurona depende de la función de activación que se le defina a la neurona. La función de activación permite mapear el resultado de forma discreta o continua, dependiendo de la función elegida tales como tangente hiperbólica, función sigmoidea, entre otras. De esta forma, el perceptrón es capaz de crear un límite de separación en las regiones dependiendo de la dimensionalidad de sus componentes. Es decir, es capaz de clasificar datos linealmente separables.

2.5.2. Funciones de activación

Mapear los datos de entrada a la salida correspondiente es la función principal de todas las funciones de activación en todos los tipos de redes neuronales. Esto significa que la función de activación [9] es la que decide si la neurona se activa a partir de la suma ponderada de las entradas al determinar el output correspondiente. Las funciones de activación más utilizadas en redes neuronales son:

- **Función sigmoidea:** se le ingresa a la función números reales transformándolos a una escala entre 0 y 1. La función sigmoidea tiene una curva en forma de S y puede ser representada matemáticamente por

$$f(x)_{sigm} = \frac{1}{1 + e^{-x}} \quad (2.3)$$

- **Función softmax:** calcula la distribución de probabilidades de cada clase respecto a todas las clases diferentes. El rango será de 0 a 1, y la suma de todas las probabilidades será igual a uno. La función Softmax está definida como

$$f_i(\mathbf{x})_{softmax} = \frac{e^{x_i}}{\sum_{j=1}^n e^{x_j}} \quad (2.4)$$

- **Función tangente hiperbólica:** Su entrada corresponde a números reales, los cuales van a ser transformados dentro del rango de -1 a 1. Su representación matemática es

$$f(x)_{tanh} = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{2}{1 + e^{-2x}} - 1 \quad (2.5)$$

- **ReLU:** es la función de activación más utilizada para redes convolucionales. Si los valores son positivos, activa la función. De otra forma el valor resultante es cero. La función ReLU está definida por

$$f(x)_{ReLU} = \max(0, x) \quad (2.6)$$

Ocasionalmente, ocurren ciertos problemas al utilizar ReLU. Por ejemplo, cuando las entradas son negativas o cero, el gradiente se vuelve cero por lo que la red no

puede realizar una propagación hacia atrás, *backpropagation* (ver sección 2.5.10), y no puede aprender. Las neuronas en esta situación se dice que están 'muertas' y no harán que la neurona se active nuevamente.

- **ELU:** es una función de activación denominada 'Unidad Lineal Exponencial'. A diferencia de las ReLU, las ELU tienen valores negativos, lo que les permite acercar las activaciones de unidades medias a cero, como la normalización por lotes, pero con una menor complejidad computacional. Los cambios medios hacia cero aceleran el aprendizaje al acercar el gradiente normal al gradiente natural de la unidad. Las ELU se saturan a un valor negativo con entradas más pequeñas y, por lo tanto, disminuyen la variación y la información propagadas hacia adelante. Está definida por

$$f(x)_{eLU} = \begin{cases} \alpha \cdot (e^x - 1) & \text{si } x \leq 0 \\ x & \text{si } x > 0 \end{cases} \quad (2.7)$$

- **LeakyReLU:** es utilizada para solucionar el problema de las neuronas 'muertas' por ReLU. En vez de que los valores negativos sean convertidos a cero, se asegura que dichos valores sean representados. Esta función se define como

$$f(x)_{LeakyReLU} = \begin{cases} \alpha \cdot x & \text{si } x < 0 \\ x & \text{si } x \geq 0 \end{cases} \quad (2.8)$$

Siendo α el factor 'Leaky' (de fuga o pérdida), con $\alpha > 0$.

2.5.3. Perceptrón multicapa

Los perceptrones multicapa (MLPs, por sus siglas en inglés *Multi-layer perceptrons*) o redes neuronales 'vainilla' [6] son los tipos más simples y comunes de redes neuronales artificiales. Pertenecen a la clase de redes neuronales conocidas como *feed-forward*, que conectan neuronas de una capa con la siguiente sin formar ciclos. A diferencia del perceptrón simple, el perceptrón solo es capaz de manejar datos linealmente separables. El perceptrón multicapa viene a salvar esta limitación siendo capaz de trabajar con datos linealmente separables y no linealmente separables.

Dado que una MLP respeta las características básicas de una red neuronal artificial, lleva su estructura básica (ver figura 2.5), que consiste en una capa de entrada, una o más capas ocultas y una capa de salida, además de una función de activación y un conjunto de pesos y biases.

Otra característica de las redes MLP[6] es el 'backpropagation', una técnica de aprendizaje supervisado para entrenar las redes neuronales (ver sección 2.5.10). De forma simple, backpropagation es una forma de ajustar los pesos dentro de la red al

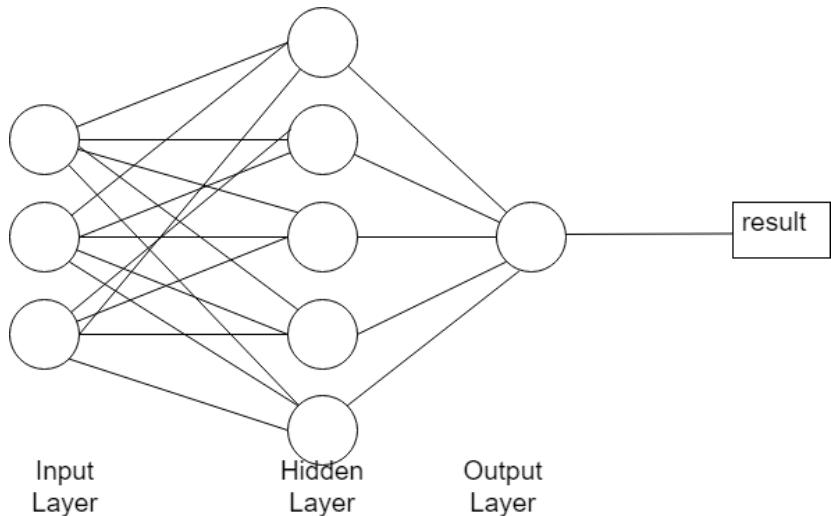


Figura 2.5: Diagrama de un perceptrón multicapa y su estructura. [6]

propagar el error del resultado final hacia atrás a la red. Esto mejora el desempeño de la red al reducir el error de la salida.

Dada su simplicidad, las MLPs necesitan períodos cortos de entrenamiento para aprender las representaciones en los datos y producir un resultado, aunque también requieren de un mayor poder de cómputo en relación al promedio, por ejemplo, a veces es necesario que el dispositivo donde se entrene lo haga a través de una GPU (Graphics Processing Unit).

Por último, la diferencia entre las MLPs y las redes neuronales profundas (DNN por sus siglas en inglés *Deep Neural Network*) es la cantidad de capas ocultas, que resulta en un tiempo más largo de entrenamiento y una mayor necesidad de poder de cómputo. Las MLPs tienen como mínimo 3 capas ocultas y, por lo general, son de menor tamaño que las redes neuronales profundas. Un ejemplo de las redes neuronales profundas, que representan el foco de este trabajo, son las redes neuronales convolucionales[6].

2.5.4. Redes Neuronales Convolucionales

Las redes neuronales convolucionales (CNN, por el término en inglés *Convolutional Neural Networks*) [9, 66] son redes neuronales artificiales profundas. La principal ventaja en comparación a sus predecesores es que detecta automáticamente las características significativas sin supervisión humana.

Las CNNs se han aplicado a un extenso rango de campos, incluyendo procesamiento de lenguaje, reconocimiento facial, *Computer vision*¹, entre otros. La estructura de las CNNs son similares a las ANNs y están basadas en el funcionamiento del cerebro animal o humano, en la corteza visual del mismo.

¹ *Computer vision* es un campo de la Inteligencia Artificial que permite a las computadoras y sistemas obtener información relevante de imágenes digitales, videos y otros medios visuales para tomar acciones o realizar recomendaciones basadas en dicha información [67].

Dentro de los beneficios de utilizar CNNs frente a otros algoritmos de Machine Learning, se puede destacar primero los pesos compartidos que reduce el número de parámetros entrenables de la red y, además, ayuda a la red a fortalecer la generalización y evitar el sobreajuste. Segundo, el aprendizaje simultáneo de las capas de extracción de características y la capa de clasificación hace que la salida del modelo esté altamente organizada y dependa en gran medida de las características extraídas.

En vez de usar capas conectadas completamente entre si (capas *fully connected*) las CNN usan conexiones locales entre neuronas, también denominado *sparse connectivity*. Por ejemplo, una neurona está únicamente conectada a las neuronas cercanas a la misma de la siguiente capa. Esto reduce significativamente el número de parámetros de la red, la cantidad de memoria requerida para almacenar dichos pesos, y el costo computacional se reduce. Más aún, todas las conexiones entre campos receptivos locales y las neuronas usan un set de pesos, al cual se denomina ***kernel***.

Un kernel o filtro es una matriz o grilla de números discretos, los cuales son los valores conocidos como pesos del filtro. Al inicio del proceso de entrenamiento, números aleatorios son asignados para actuar como los pesos del kernel, un proceso conocido como inicialización de pesos y para lo cual existen diferentes métodos. A medida que transcurre el entrenamiento, con cada ciclo, los pesos se ajustan a los datos que se le provee a la red; por lo tanto, el kernel aprende a extraer las características significativas para lo que la red es entrenada.

El kernel será compartido por todas las neuronas que están conectadas, y el resultado de los cálculos entre los campos receptivos y las neuronas serán almacenados en una matriz llamada mapa de activación. Consecuentemente, distintos kernels resultaran en distintos mapas de activación, y el número de kernels puede ajustarse a través de hiperparámetros² al entrenar la red.

De esa forma, independientemente del número de conexiones totales entre las neuronas, la cantidad de pesos totales corresponde al tamaño del kernel. Al combinar la propiedad de compartir pesos con la de conexiones locales, una CNN es capaz de manipular datos con grandes dimensiones.

2.5.5. Capas convolucionales

En una arquitectura de CNN, el componente más importante es la capa convolucional [9, 66]. De forma similar a la capa oculta de los perceptrones multicapa, estas capas tienen la función de convertir la entrada a una representación de un nivel más abstracto y consiste en una colección de filtros convolucionales. Los datos de entrada,

²Los hiperparámetros [68] son variables de configuración establecidas antes de iniciar el entrenamiento, permitiendo controlar el proceso en sí mismo de un modelo. Algunos ejemplos de hiperparámetros son la cantidad de capas ocultas de neuronas artificiales a usar entre la capa de entrada y la de salida, la función de activación, la tasa de aprendizaje, el optimizador.

expresados como métricas N-dimensionales, son transformados con los kernels a través de la convolución para generar la matriz de salida o mapa de características.

El input de la red neuronal tradicional es un vector, mientras que el input de una CNN es una imagen multicanal o incluso un video. Esta imagen multicanal se puede representar como una matriz de N dimensiones donde cada valor individual corresponde a un pixel de la imagen.

El kernel recorre toda la imagen de forma horizontal y vertical, mientras se realiza el producto escalar entre el kernel y los valores numéricos de la imagen. El producto escalar calculado representa el mapa de características de la salida. Estos mapas de activación pueden contener las características extraídas por varios kernels. Cada kernel puede actuar como un extractor de características y compartirá sus pesos con todas las neuronas a las que esté conectado. Por último los valores resultantes del producto se suman para crear un único valor escalar que corresponde a un lugar de la matriz de salida o activación. El proceso se repite hasta agotar los movimientos del kernel sobre la imagen.

Por lo general, una función de activación no lineal (ReLU, tanh, sigmoidea, entre otras) es aplicada a los valores de las operaciones entre el kernel y el input. Estos valores son almacenados en los mapas de activación, que pasarán a la siguiente capa de la red (ver figura 2.6). Para obtener el valor del pixel final (-3) de la figura, se multiplica los píxeles resaltados en verde de la imagen de entrada por los números del filtro de convolución con el producto escalar y se realiza la sumatoria de todos los valores. Como se puede ver en la figura, el cálculo realizado es:

$$(-1 \cdot 3) + (0 \cdot 0) + (1 \cdot 1) + (-2 \cdot 2) + (0 \cdot 6) + (2 \cdot 2) + (-1 \cdot 2) + (0 \cdot 4) + (1 \cdot 1) = -3 \quad (2.9)$$

Para el proceso de convolución, algunos motivos espaciales deben ser definidos para producir mapas de activación de cierto tamaño. Los atributos esenciales incluyen:

- El tamaño del kernel. Cada kernel tiene un tamaño de ventana, que también se refiere al campo receptivo. El kernel realiza una convolución en una región de la imagen con el mismo tamaño que su ventana para producir la matriz de activación.
- *Stride* o paso. Este parámetro define el número de píxeles de la imagen que el kernel se mueve para la siguiente posición donde va a realizar la convolución. Si se establece un paso de 1, una vez que se realice la convolución con el kernel, el mismo se desplaza al pixel inmediato de al lado para realizar la siguiente operación, y así sucesivamente hasta alcanzar los bordes. El *stride* puede definirse tanto en ancho como en el largo de la imagen y, de ser un video, en la dimensión temporal. No obstante, hay que considerar que cuánto mayor sea el stride, más pequeño será

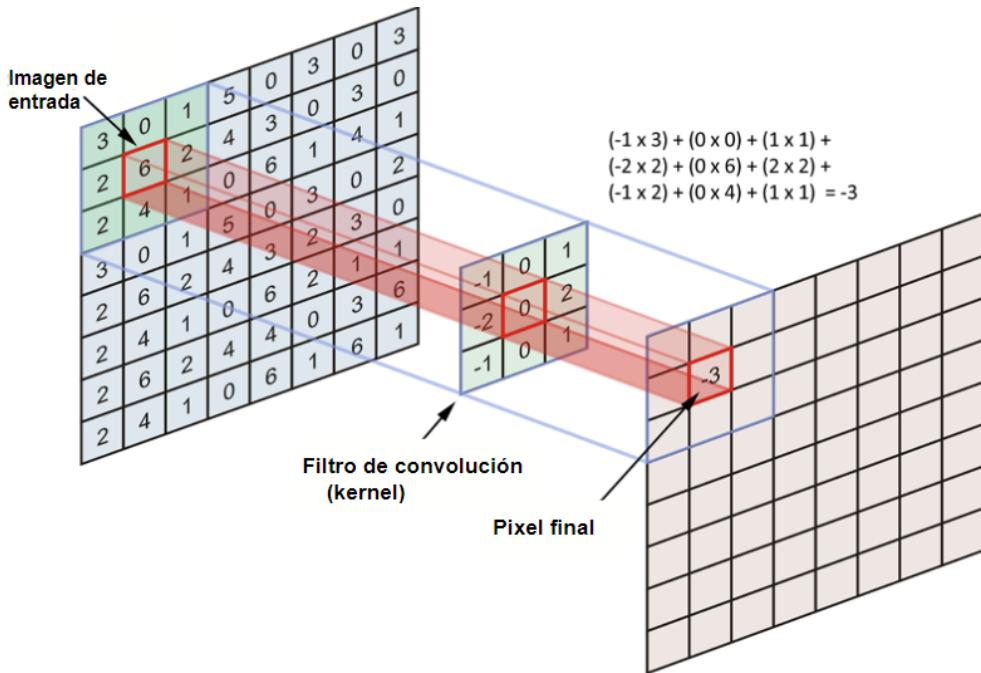


Figura 2.6: Diagrama de proceso de convolución [7]. Se toma una imagen de un solo canal representada por una matriz de 8x8 (matriz azul) con un kernel de 3x3 (verde).

el mapa de activación resultante.

- **Zero-padding.** Este parámetro es usado para especificar con cuántos ceros se va a llenar la imagen en sus bordes. Esto es útil para preservar la dimensión del input en la convolución.

Estos tres parámetros son los más conocidos e implementados para controlar el volumen de la salida de una capa convolucional. Específicamente, para una imagen de entrada de dimensiones $W_{input} \times H_{input} \times Z$ (ancho, alto, y cantidad de canales de la imagen, respectivamente), con los hiperparámetros del tamaño del kernel (N), stride (S) y zero-padding (P), las dimensiones del mapa de activación se calculan tales que

$$W_{output} = \frac{(W_{input} - N + 2P)}{S} + 1, \quad (2.10)$$

$$H_{output} = \frac{(H_{input} - N + 2P)}{S} + 1, \text{ y} \quad (2.11)$$

$$D = Z \quad (2.12)$$

Siendo W_{output} el ancho de la matriz de activación; H_{output} la altura de la matriz de activación y D la cantidad de canales de la matriz.

En el caso de la figura 2.6, al no tener zero-padding y asumiendo un stride de 1, se puede concluir que la matriz resultante resulta con dimensiones 6 x 6 x 1. Esto se debe

a que la altura y el ancho tienen las mismas dimensiones, de forma que se obtiene

$$H_{output} = W_{output} = \frac{(8 - 3 + 2 * 0)}{1} + 1 = 6 \quad (2.13)$$

De esta forma, la matriz de activación se reduce de tamaño en comparación a la imagen original. Una posible solución a esto es realizarle zero padding a la imagen, rellenando los bordes con ceros, aumentando así la dimensión de la imagen original.

Un ejemplo está ilustrado en la figura 2.7. En el mismo, la imagen original tiene dimensiones de $5 \times 5 \times 1$ de ancho, largo y canales respectivamente; el kernel tiene dimensiones de 3×3 ; el stride es de 1 y el zero-padding realizado también corresponde a 1. El tamaño de la matriz de salida, en este caso, no se modifica al aplicarle el filtro, permaneciendo con dimensiones de $5 \times 5 \times 1$.

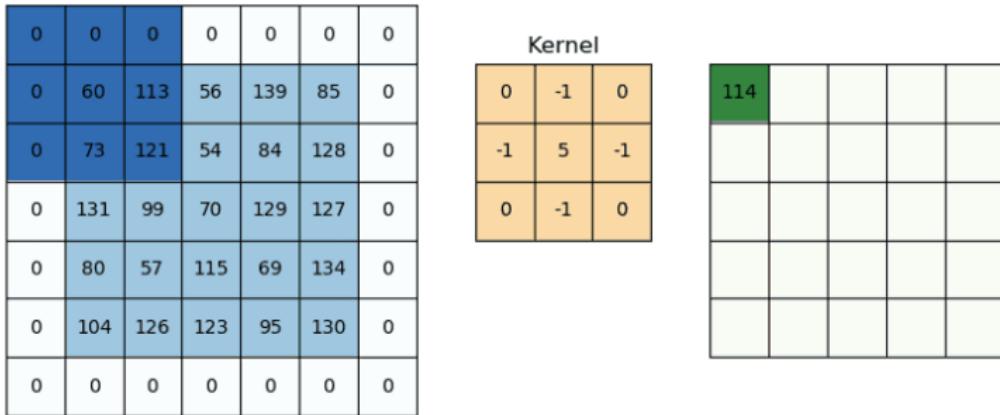


Figura 2.7: Convolución con zero-padding. De izquierda a derecha: imagen de entrada, kernel y matriz de activación [8].

2.5.6. Capas de pooling

Las capas de pooling [9, 66], por lo general, se ubican entre dos capas convolucionales. El objetivo de las mismas es submuestrear los mapas de características creados por las operaciones convolucionales, es decir, que reducen de dimensión el input, conservando la mayor cantidad de información posible. Además, las capas de pooling son capaces de introducir invarianza espacial a la red neuronal que puede ayudar a mejorar la generalización del modelo.

De forma similar a la convolución, se debe asignar un stride o paso, zero-padding y un tamaño de ventana dado por un kernel como hiperparámetros. La capa de pooling escanea el input completo a través de la ventana/kernel de la misma manera que el kernel convolucional.

Entre las capas de pooling, se pueden definir subtipos como max pooling, averaging pooling, global average pooling, entre otros (ver figura 2.8). Cada uno de estos tipos se diferencia en el método por el cual realizan la reducción de dimensiones.

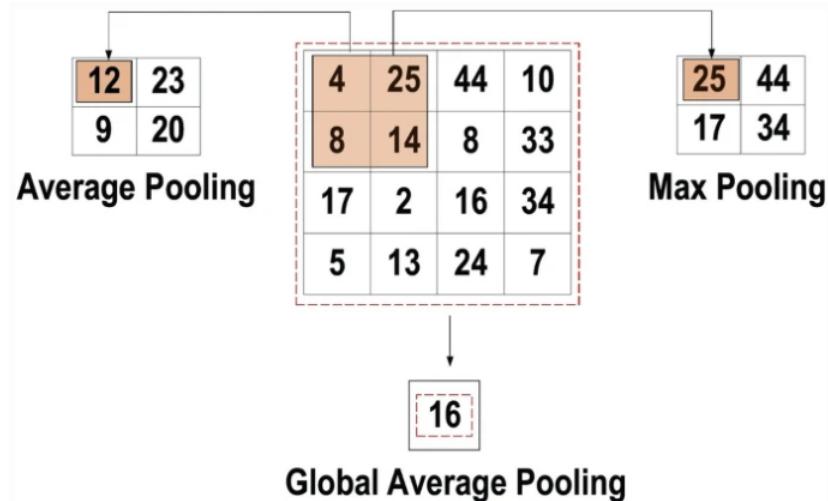


Figura 2.8: Ejemplos de tipos de pooling [9]. Se tiene una matriz de entrada de 4×4 y el pooling usará un kernel de 2×2 . Por lo tanto, la matriz de salida tendrá dimensiones de 2×2 , en el caso de average y max pooling. En el average pooling, cada valor de la salida será un promedio de los valores de la ventana; mientras que con el max pooling, el valor final será el máximo de aquellos que se encuentren en la ventana.

2.5.7. Sobreajuste

El principal problema para los modelos de redes convolucionales es el overfitting o sobreajuste. Un modelo tiene sobreajuste cuando el modelo obtiene buenos resultados sobre los datos de entrenamiento pero no en datos de testeo. El modelo pierde generalización de datos que no 'ha visto' para obtener las características que permiten la clasificación o regresión. De forma contraria, un modelo subajustado no ha aprendido lo suficiente de los datos de entrenamiento como para tener buenos resultados, tanto para los datos de entrenamiento como para el de prueba.

Para solucionar el problema del sobreajuste, existen técnicas de regularización [9, 69], que consisten en penalizar de alguna forma las predicciones que hace la red durante el entrenamiento, de forma que no tome al dataset de entrenamiento como verdad absoluta y generalice mejor frente a otros datos. Algunos ejemplos de estas técnicas son las capas de Drop-out, el método de Drop-weights, Data Augmentation, Batch normalization. Entre ellas, se utilizaron:

1. Capa de Drop-out: En cada epoch en el entrenamiento, algunas neuronas son omitidas aleatoriamente. Debido a esto, el poder de seleccionar características es distribuido igualmente a través del resto de las neuronas, al mismo tiempo que fuerza al modelo a aprender características diferentes e independientes entre si.

Durante el entrenamiento, las neuronas omitidas no serán parte de la propagación. Al terminar el entrenamiento, de todos modos se utilizarán todas las neuronas para realizar la predicción en los datos de validación y prueba.

2. Batch Normalization: este método ayuda a normalizar la salida del modelo y ayuda a reducir el cambio interno de la covarianza de las capas de activación. Este cambio se vuelve cada vez más alto a medida que se ajustan los pesos en el entrenamiento, que puede ocurrir si los datos de entrenamiento se obtienen de numerosas fuentes distintas entre sí. Por lo tanto, el modelo tardará aún más en converger y, por lo tanto, su entrenamiento será más largo. Entonces, las ventajas de aplicar 'Batch Normalization' son las siguientes:

- Previene el problema de desvanecimiento de gradiente³.
- Puede controlar una inicialización de los pesos pobre.
- Reduce significativamente el tiempo para la convergencia, especialmente para los datasets grandes.
- Se esfuerza por disminuir la dependencia del entrenamiento a través de hiperparámetros.
- Se reduce la probabilidad de que ocurra un sobreajuste del modelo.

2.5.8. Capas densas o fully-connected

Comúnmente, estas capas están situadas al final de la arquitectura de la red convolucional. En este tipo de capa, cada neurona está conectada a las neuronas de la capa anterior, por eso se la llama capa densa o fully-connected (FC). Sigue los métodos de la red MLP (ver sección 2.5.3) dado que es un tipo de red neuronal feed-forward.

La entrada de la capa viene de la última capa de pooling o capa convolucional dependiendo de la arquitectura. Este input es un vector que es creado a partir de mapas de características después del flattening⁴ (ver figura 2.9). El resultado de la última capa densa es el resultado final de la red convolucional.

2.5.9. Funciones de pérdida

La predicción final es realizada por la última capa dentro de la arquitectura de la red convolucional. Con ese valor, la función de pérdida [9] calcula el error de la predicción en comparación al valor de la etiqueta en el entrenamiento. Este error mide

³Cuanto más compleja y más capas tenga una red, los gradientes de las funciones de pérdida se hacen cercanos a cero, y las redes son difíciles de entrenar. Para más información sobre las funciones de pérdida ver sección 2.5.9

⁴Las capas de flattening son aquellas que transforman un vector de N dimensiones a un vector unidimensional para que pueda ser utilizado por una capa densa [7].

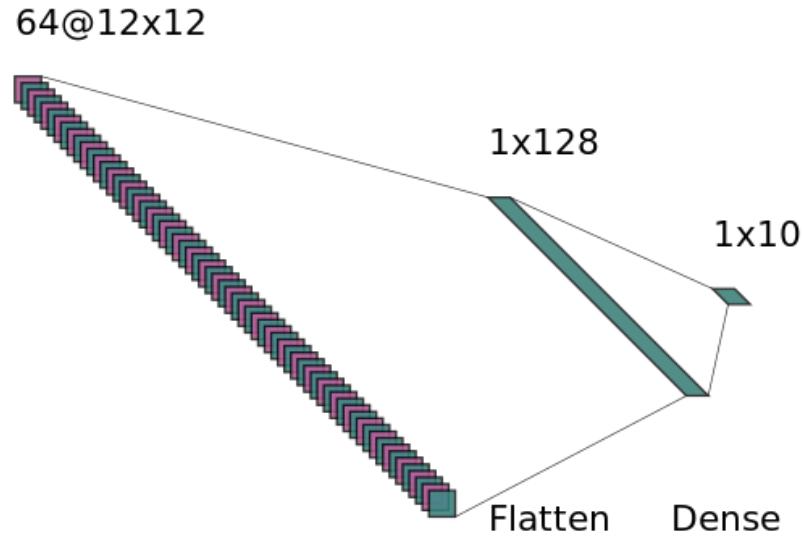


Figura 2.9: Modificaciones dimensionales del vector output tras pasar por capa densa y flattening [7].

que tan bueno es el desempeño del modelo en relación al objetivo y ayuda a optimizar el modelo en el proceso de aprendizaje y entrenamiento.

Hay diversos tipos de funciones de pérdida utilizados de acuerdo al problema que se presente. Algunos ejemplos son la función Softmax, la función de pérdida euclídea, la función de pérdida de la bisagra, entre otras. Particularmente, en los modelos del proyecto se utilizaron:

1. **Función de entropía cruzada o Softmax:** La salida de esta función es una probabilidad entre 0 y 1. Es utilizada para sustituir a la función del error cuadrático en problemas de clasificación multiclas. En la capa de salida, se utiliza una capa de activación Softmax para generar una clase a partir de la distribución de probabilidad. La probabilidad de la clase de salida i puede calcularse como

$$p_i = \frac{e^{a_i}}{\sum_{k=1}^N e^a_k}. \quad (2.14)$$

En esta, e^{a_i} representa la salida no normalizada de la capa precedente, mientras que N representa el número de neuronas en la capa de salida. Finalmente, la entropía cruzada se calcula como

$$H(p, y) = - \sum_i y_i \cdot \log(p_i) \quad \text{where } i \in [1, N] \quad (2.15)$$

donde la predicción se denota p_i y la salida objetivo, y_i .

2. **Función de pérdida euclídea:** Es utilizada ampliamente para problemas de

regresión, y es también denominada el error medio cuadrático. Se calcula como

$$H(p, y) = \frac{1}{2N} \sum_{i=1}^N (p_i - y_i)^2. \quad (2.16)$$

3. Otras funciones pueden ser utilizadas como funciones de pérdida para mejorar el desempeño de los modelos. Por ejemplo, en uno de los modelos desarrollados en este trabajo, se utilizó **la función de correlación de Pearson** para comparar señales de fotopletismografía. El coeficiente de Pearson (ρ) es una medida estadística que indica la relación entre dos variables. Se calcula como

$$\rho_{XY} = \frac{T \cdot \sum_{i=1}^T x_i y_i - (\sum_{i=1}^T x_i) \cdot (\sum_{i=1}^T y_i)}{\sqrt{(T \cdot \sum_{i=1}^T x_i^2 - (\sum_{i=1}^T x_i)^2)(T \cdot \sum_{i=1}^T y_i^2 - (\sum_{i=1}^T y_i)^2)}} \quad (2.17)$$

donde T es el largo de las señales, x es la señal original (en este caso, la señal de fotopletismografía), e y es la predicción de la señal realizada por el modelo.

El objetivo del entrenamiento es reducir el error o aumentar la exactitud o precisión al cambiar los pesos de las neuronas, convergiendo al mínimo o máximo valor posible, respectivamente. Para esto son complementados por técnicas basadas en gradiente.

2.5.10. Optimizadores y back-propagation

Como se ha mencionado, para que la red 'aprenda', los parámetros de las redes neuronales deberían actualizarse a lo largo de las epochs del entrenamiento. Al mismo tiempo, la red debería buscar por la respuesta local óptima en el entrenamiento para minimizar el error.

Con ese objetivo, el algoritmo de descenso de gradiente [9] se encarga de modificar los parámetros de la red en cada epoch. Para realizarlo correctamente, necesita computar el gradiente de la función objetivo aplicando una derivada de primer orden respecto a los parámetros de la red. Luego, el parámetro es actualizado en dirección contraria a este gradiente para reducir el error, aumentando o disminuyendo los pesos de las neuronas. La actualización es realizada a través de lo que se llama 'back-propagation'. En este proceso el gradiente de cada neurona se propaga hacia atrás a todas las neuronas de la capa anterior. Esta operación se representa matemáticamente como

$$w_{ij}^t = w_{ij}^{t-1} - \Delta w_{ij}^t, \quad \Delta w_{ij}^t = \eta \cdot \frac{\delta E}{\delta w_{ij}}. \quad (2.18)$$

donde w_{ij}^t es el peso final del epoch actual (t), w_{ij}^{t-1} es el mismo peso en el epoch t-1, η es el learning rate, y E es el error de la predicción. El learning rate es definido como el tamaño del paso en el parámetro actualizado. Un epoch representa una repetición

completa de la actualización que involucra al dataset de entrenamiento completo en una vez. Por lo tanto, seleccionar el learning rate correcto es necesario para que este influya en el proceso de aprendizaje.

Hay distintas alternativas para realizar un algoritmo basado en gradiente que son comúnmente utilizados, por ejemplo: el descenso del gradiente en batch o en mini-batch, el descenso del gradiente estocástico, momento o estimación de momento adaptativo (Adam). En este trabajo se utilizó el algoritmo Adam, el cual es representado por la matriz hessiana, que utiliza derivadas de segundo orden. Ha sido diseñado específicamente para entrenar redes neuronales profundas de forma más eficiente en cuestiones de memoria y capacidad de cómputo. El mecanismo de Adam calcula un learning rate adaptativo para cada parámetro del modelo, usando gradiente cuadráticos para escalar el learning rate y el promedio móvil del gradiente. Su ecuación es

$$w_{ij^t} = w_{ij^{t-1}} - \frac{\eta}{\sqrt{\widehat{E[\delta^2]}^t + \epsilon}} \cdot \widehat{E[\delta^2]}^t \quad (2.19)$$

2.5.11. Redes convolucionales 2D

Las CNN 2D han demostrado un rendimiento sobresaliente en la clasificación de imágenes. Estas redes son capaces de analizar las características visuales de una imagen mediante la aplicación de filtros convolucionales lo que les permite capturar patrones relevantes en los datos. Además, las CNN 2D suelen lograr altas precisiones al clasificar categorías que son distinguibles para el ojo humano, como es este proyecto. Esto se debe a su capacidad para detectar y aprender características visuales específicas, como bordes, texturas y formas.

Actualmente existen numerosas arquitecturas de redes neuronales 2D ampliamente estudiadas y utilizadas en el campo de la visión por computadora. Ejemplos destacados de estas arquitecturas que fueron utilizadas en este trabajo incluyen ResNet [70] y VGG [71]. Estas redes han sido entrenadas utilizando conjuntos de datos masivos, como ImageNet [72], y han logrado alcanzar altas precisiones en la clasificación de imágenes, llegando a valores de del 0.9 de exactitud en tareas de clasificación multiclas. Por lo tanto, a dichos modelos se les puede aplicar Transfer Learning, o transferencia de aprendizaje. La transferencia de aprendizaje implica utilizar los conocimientos adquiridos por una red preentrenada, que ha aprendido a reconocer una amplia gama de características visuales en imágenes generales, y aplicar estos conocimientos a una tarea específica. En el caso de las CNN 2D, las primeras capas convolucionales suelen aprender filtros que identifican patrones generales, como líneas y manchas, mientras que las últimas capas se especializan en la identificación de características más específicas según la tarea que se les ha asignado. Esto significa que se puede cargar una red preentrenada, como ResNet o VGG, que ha sido entrenada con el conjunto de datos

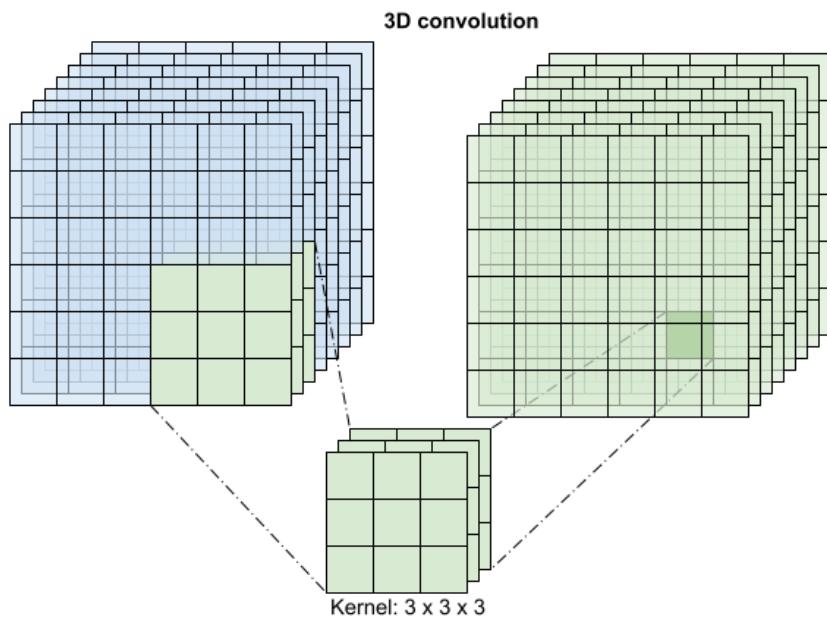


Figura 2.10: Convolución de 3 dimensiones aplicada a videos [10].

masivo de ImageNet, y ajustar únicamente los pesos de las capas finales durante el entrenamiento con imágenes propias para la tarea deseada. De esta manera, se aprovecha el conocimiento general y las características aprendidas por la red preentrenada, lo que puede mejorar significativamente los resultados incluso cuando se disponen de un número limitado de imágenes para entrenar un modelo.

2.5.12. Redes convolucionales 3D

Los videos pueden definirse como una secuencia de imágenes. Es decir que sus 3 dimensiones serían 2 espaciales (el ancho y alto de la imagen) y el tiempo que dura el video. Además a esto, se le suma que cada imagen tiene 3 canales de colores. Siguiendo este pensamiento, es posible aplicar redes convolucionales a datos volumétricos como lo serían los videos y es por esto que se crean las redes convolucionales 3D [73–75].

Las redes convolucionales 3D son capaces de preservar la información temporal y propagarla a través de las capas de la red. Esto lo realiza a través de la convolución que es capaz de afectar 3 de las 4 dimensiones nombradas del video. Dado que se asume que la cantidad de canales a la entrada será la misma que a la salida, de la misma forma que ocurre con las imágenes en las redes convolucionales 2D, el kernel es capaz de moverse en tiempo del video y, altura y ancho de la imagen del video (ver figura 2.10).

No obstante, las redes convolucionales 3D introducen gran cantidad de parámetros al modelo, por lo cual inevitablemente requerirán gran cantidad de datos, tiempo de entrenamiento y capacidad de cómputo para generar buenos resultados. Además se

debe considerar que la mayoría de los métodos de modelos 3D solo aceptan como entrada clips cortos y exploran un atributo en específico del video.

2.6. Evaluación del modelo

Existen diversas métricas para la evaluación del rendimiento de los modelos de regresión. Entre las típicas, se pueden encontrar el RMSE y MAE, los cuales fueron utilizados en este trabajo para comparar los resultados de cada enfoque y así medir el rendimiento de los mismos.

Además, como métrica secundaria, se calcularon el coeficiente de determinación en ciertos modelos, o la correlación de Pearson (que también se utilizó como función de perdida) como medida de correlación entre etiquetas o datos reales y las predicciones realizadas.

2.6.1. Raíz del error cuadrático medio (RMSE)

La raíz del error cuadrático medio es el desvío estándar de los residuos. Los residuos son una medida de que tan lejos está la línea de regresión predicha de los puntos de datos originales. Es la distancia entre dicha línea de mejor ajuste y los datos. Se calcula como

$$RMSE = \sqrt{\frac{\sum_{i=1}^T (x_i - y_i)^2}{T}} \quad (2.20)$$

donde T es el largo de las señales, x es la señal de fotopletismografía original, e y es la predicción de la señal de fotopletismografía realizada por el modelo [76] [77].

2.6.2. Error absoluto medio (MAE)

El error absoluto medio proporciona el promedio de la diferencia absoluta entre la predicción del modelo y el valor objetivo. Se calcula como

$$MAE = \frac{\sum_{i=1}^T |x_i - y_i|}{T} \quad (2.21)$$

donde T es el largo de las señales, x es la señal de fotopletismografía original, e y es la predicción de la señal de fotopletismografía realizada por el modelo [78].

2.6.3. Coeficiente de Pearson (ρ)

El coeficiente de correlación de Pearson (ver definición en sección 2.5.9) trata de cuantificar la dependencia lineal entre dos variables aleatorias cuantitativas. La corre-

lación de Pearson puede tomar valores entre -1 y 1, incluidos. Un valor de cero implica que no hay asociación lineal entre las dos señales, mientras que una correlación de 1 o -1 indica una fuerte correlación positiva o negativa respectivamente[79] [80].

2.6.4. Coeficiente de Determinación (R^2)

El coeficiente de determinación es un valor entre 0 y 1 que mide qué tan bien un modelo de regresión se aproxima a los datos reales [81]. Sirve para cuantificar el poder predictivo de un modelo de regresión. Más específicamente, es la proporción de la varianza en la variable dependiente que es explicada por el modelo [82]. Se calcula como

$$1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y}_i)^2} \quad (2.22)$$

donde y_i es el valor real de la muestra i, \hat{y}_i es el valor predicho de la muestra i, y \bar{y}_i es el promedio de los valores reales.

Capítulo 3

Materiales y métodos

3.1. Lenguaje de programación

Todo el desarrollo del proyecto planteado se llevó a cabo utilizando el lenguaje de programación Python. Este lenguaje se utilizó para hacer todos los procesamientos de videos y señales, como también para la construcción, entrenamiento y evaluación de todos los modelos desarrollados.

Python es un lenguaje de programación de alto nivel, interpretado, con una filosofía de diseño que se enfoca en la legibilidad del código y la simplicidad. Fue lanzado por primera vez en 1991. En los últimos años, Python ha adquirido una relevancia significativa en el campo tecnológico debido a su capacidad para trabajar de manera sencilla con campos avanzados como aprendizaje automático, procesamiento de grandes volúmenes de datos (Big Data), inteligencia artificial y ciencia de datos[83].

En el contexto de este trabajo, Python[83, 84] se elige como el lenguaje de programación debido a los siguientes beneficios:

- Legibilidad y facilidad de uso: Python destaca por su sintaxis clara y expresiva, lo que facilita la lectura y comprensión del código, reduciendo el tiempo necesario para el desarrollo y la depuración. Esto es especialmente valioso para equipos de investigación y desarrollo, ya que permite una colaboración efectiva y un proceso ágil de prototipado.
- Amplio ecosistema de bibliotecas y frameworks: Python cuenta con una gran cantidad de bibliotecas y frameworks especializados en *Computer vision* y aprendizaje profundo, como TensorFlow, PyTorch, Keras y OpenCV. Estas herramientas permiten la implementación eficiente de algoritmos complejos y aceleran el proceso de desarrollo de soluciones biomédicas avanzadas.
- Comunidad activa y soporte: La comunidad de desarrolladores de Python es activa y comprometida, lo que resulta en un soporte constante, abundante documen-

tación y una amplia gama de recursos disponibles en línea. Esta colaboración fomenta la adopción de las últimas técnicas y avances en *Computer vision* y aprendizaje profundo, lo que es esencial para mantener la competitividad en la investigación biomédica.

- Portabilidad y facilidad de integración: Python es compatible con varias plataformas y sistemas operativos, lo que facilita la implementación de soluciones en diferentes entornos y la integración con otros lenguajes de programación y bibliotecas optimizadas, como C++ y CUDA, para el aprovechamiento de aceleradores de hardware y el rendimiento óptimo en tareas intensivas de cómputo.

3.2. Enfoques

El objetivo del este proyecto fue desarrollar un modelo para la plataforma de Úma Health que pudiese extraer signos vitales (en esta primera iteración, frecuencia cardíaca) a partir de la señal de FPG obtenida a través de un vídeo capturados con dispositivos personales. Para cumplir este objetivo se desarrollaron dos enfoques:

- Utilizar videos del dedo índice cubriendo completamente la cámara (FPG de contacto).
- Utilizar 'video-selfies' o videos de la cara del sujeto (FPG remota).

Cada uno de estos enfoques presentó distintas ventajas y desventajas. Para empezar, existen múltiples investigaciones centradas en FPG de contacto las cuales demostraron sus resultados y factibilidad; y la amplia experiencia del campo en este desarrollo facilitó su implementación. En cambio, la FPG remota es una solución más innovadora que aún no ha sido estudiada en profundidad.

Otra gran diferencia entre ambos métodos fue la relación señal-ruido. En la FPG de contacto únicamente se captura un área en donde la señal de FPG es fuerte. Por ende, la relación señal ruido es mejor que en la FPG remota en la que el sujeto se encuentra a una distancia del sensor. Además, una señal de contacto se toma fácilmente en condiciones de poca iluminación, utilizando un flash, mientras que el resultado de una FPG remota varía mucho con las condiciones de iluminación.

Por otro lado, la captura de una FPG remota resulta mucho más intuitiva para cualquier usuario que una FPG de contacto, y es más amigable en el contexto de una consulta virtual. Más aún, con el algoritmo desarrollado para los videos de la cara, se podría potencialmente medir otros signos vitales o características de la persona en simultáneo, mientras que este potencial fue más acotado para videos del dedo debido a los datos disponibles.

Cada enfoque presenta un desarrollo en sí mismo con su propio uso de los datasets, modelos¹, procesamiento y resultados. Por esta razón, los métodos y resultados se dividieron en dos, una sección por cada enfoque, para mayor claridad.

3.3. Datasets empleados

Dataset	Origen	Contenido	Etiqueta asociada	Rango (latidos por minuto)	Enfoque	Modelos	Etapa
UMA-dPPG	Propio	Videos del dedo índice apoyado contra la cámara	Valor escalar de frecuencia cardíaca	51 a 159	FPG de contacto	1DCNNv1.X, 1DCNNv2.X	Entrenamiento y validación
UMA-dPPG-extensión	Propio	Videos del dedo índice apoyado contra la cámara	Valor escalar de frecuencia cardíaca	52 a 153	FPG de contacto	1DCNNv2.X	Entrenamiento y validación
Dataset de testeo	Propio	Videos del dedo índice apoyado contra la cámara	Valor escalar de frecuencia cardíaca	62 a 162	FPG de contacto	1DCNNv1.3, 1DCNNv1.4, 1DCNNv1.6, 1DCNNv2.4	Test
UMA-rPPG	Propio	Videos de la cara	Valor escalar de frecuencia cardíaca	50 a 170	FPG remoto	3DCNNv1.X	Entrenamiento
UMA-rPPG	Propio	Videos de la cara con magnificación eulariana	Valor escalar de frecuencia cardíaca	50 a 150	FPG remoto	3DCNNv2.X	Entrenamiento
UBFC-rPPG	Público	Videos de la cara	Valor escalar de frecuencias cardíacas estimadas a partir de la FPG asociada al video	40 a 180	FPG remoto	3DCNN v2.X	Test
UBFC-rPPG	Público	Videos de la cara	Valor escalar de frecuencias cardíacas estimadas a partir de la FPG asociada al video	40 a 100	FPG remoto	3DCNN v1.X y v2.X	Test acotado
UBFC-rPPG	Público	Videos de la cara	señales de FPG	40 a 180	FPG remoto	3DCNN v3	Entrenamiento
UMA-rPPG	Propio	Videos de la cara	Valor escalar de frecuencia cardíaca	50 a 170	FPG remoto	3DCNNv3	Testeo del flujo completo
UMA-rPPG	Propio	Videos de la cara	Valor escalar de frecuencia cardíaca	77 a 135	FPG remoto	3DCNNv3	Testeo acotado del flujo completo

Tabla 3.1: Especificaciones de los datasets utilizados en el desarrollo del algoritmo relacionados a los modelos en los que fueron utilizados.

Con el fin de entrenar y testear los algoritmos generados, se utilizaron diversas bases de datos. Considerando la falta de datos necesarios, en principio, se conformaron tres datasets propios (Uma-dPPG, Uma-dPPG-extensión y Uma-rPPG) en las oficinas de Üma Health los cuales fueron complementados posteriormente con datasets públicos obtenidos en internet (UBFC-rPPG[85]).

Estas bases de datos se clasificaron de acuerdo al tipo de video (enfoques) y etiqueta asociada al mismo que corresponde a la predicción que realizaría el modelo. De esta

¹Los modelos siguen la notación NombreDeModelo.vY.X, siendo X e Y números enteros. El número representado por Y corresponde a la versión del modelo con una estrategia particular, mientras que la X es la versión de dicha estrategia con cambios en los hiperparámetros durante los entrenamientos.

forma, se definieron 3 tipos de modelos diferentes: predicción de la frecuencia cardíaca a partir de FPG de contacto, predicción de la frecuencia cardíaca a partir de FPG remota y predicción de la señal de FPG a partir de FPG remota (ver tabla 3.1). Cada uno de los datasets presentó diversas características en el momento de la toma de los videos y diversas distribuciones de las frecuencias cardíacas que se utilizaron como etiquetas por lo cual se realizaron ajustes en el preprocesamiento y evaluación en la etapa de testeo de modelo correspondiente.

3.3.1. Datasets generados en Üma Health

Para la obtención de los videos que conformaron los datasets propios, se realizó un protocolo alineado a la normalización de las condiciones y la comparación con un oxímetro de pulso comercial.

Los siguientes elementos fueron necesarios para realizar la toma de mediciones:

- Cronómetro
- Cinta métrica
- Trapo y líquido limpia vidrios
- Trípode para toma facial
- 2 teléfonos celulares especificados en las secciones 3.3.1 y 3.3.1
- Oxímetro comercial

Método de medición

La obtención de las mediciones se realizó dentro de las oficinas de Üma Health. 40 miembros del personal que se encontraban en el momento de realizar el estudio fueron voluntarios para las mediciones, los cuales dieron su consentimiento firmado.

Previo a comenzar la toma de video, se registraron las siguientes características del sujeto en una planilla de Excel la cual fue guardada en el repositorio de la empresa:

- Número identificatorio de sujeto
- Fecha
- Sexo de nacimiento
- Edad
- Barba (Sí/No)

- Anteojos (Sí/No)
- Caso - en reposo (Sí/No)
- Caso - tras actividad física (Sí/No)
- Caso - Apnea (Sí/No)
- Frecuencia cardíaca de oxímetro
- Saturación de oxígeno de oxímetro
- Código de nombre de archivo autogenerado

Las capturas de video se iniciaron 5 segundos antes de una señal sonora, para luego poder ser recortadas en el momento en el que se escuchase la misma. Dicha señal sonora se realizó al momento de iniciar el cronómetro, luego de la calibración del oxímetro (5s).

Las capturas de video de tomaron con una duración de 35 s tanto para la toma de la cara como la del dedo. Además se registraron mediciones con un oxímetro de pulso comercial Coronet, Seiseme, el cual se sincronizó con los dispositivos de video al recortar los mismos. El oxímetro de pulso proveyó los valores medios de la frecuencia cardíaca en latidos por minuto (lpm) y la saturación de oxígeno en porcentaje. El dispositivo tiene un sensor LED de doble longitud de onda. El rango de medición para la saturación de oxígeno es 35 a 100 % con una exactitud $\pm 3\%$. En cuanto a la frecuencia cardíaca, su rango va entre 30 lpm y 240 lpm con una exactitud de ± 3 lpm.

El oxímetro de pulso se colocó en la mano izquierda (ver figura 3.1) en el dedo índice con una presión suficiente como para evitar artefactos, pero no demasiada como para cortar el flujo sanguíneo. Se verificó que el sujeto no tuviese las uñas pintadas.

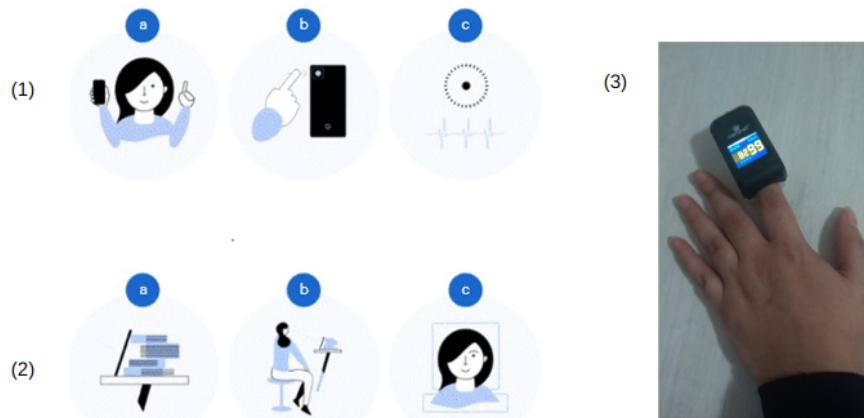


Figura 3.1: Diagramas de medición de las muestras. (1) En la mano derecha del sujeto se coloca el dedo cubriendo la cámara y el flash [11], (2) se coloca la cámara a cierta distancia del sujeto de forma que se encuadre el rostro [11] y (3) en la mano izquierda se coloca el oxímetro de pulso comercial para obtener las etiquetas.

Se monitorizó el oxímetro durante todas las capturas de videos y se anotaron tres mediciones de saturación de oxígeno y frecuencia cardíaca para cada captura: una a los 10 segundos de medición, una a los 20 segundos de medición y una a lo 30 segundos de medición. Para que cada medición fuese considerada válida, esta debió mantenerse constante durante tres latidos seguidos. En caso contrario, se tomó un promedio de las mediciones de los tres latidos consecutivos. La medición con el oxímetro se realizó una vez por cada sujeto, sincrónicamente con los dos videos.

Descripción de casos

Se tomaron los videos y mediciones de oxímetro en 3 situaciones para cada sujeto:

- En reposo: Los sujetos se sentaron frente a la cámara con las condiciones de no haber realizado actividad física; no haber tomado ni té, ni café, ni mate; no haber fumado en los últimos 15 minutos. Esta fue la primera medición realizada.
- Apnea: Los sujetos permanecieron sentados en una silla y mantuvieron la respiración durante el tiempo suficiente como para causarle una molestia. Desde el inicio de la apnea, se inició la grabación y las mediciones con el oxímetro, generando una marca sonora a partir de que los sujetos volvían a respirar. En el oxímetro se tomó el valor más bajo registrado, de acuerdo a lo establecido en la sección 3.3.1. Luego de esta prueba los sujetos debieron mantenerse en reposo por 2 minutos para volver a su saturación y frecuencia normal antes de proceder a la siguiente prueba.
- Actividad física: Luego de las mediciones con apnea los sujetos eligieron realizar 20 sentadillas con salto o bajar y subir escaleras para aumentar su frecuencia cardíaca. Al finalizar la actividad seleccionada se sentaron en una silla, se les colocó el oxímetro y se procedió a tomar los videos. Se descartaron los primeros 5 segundos de los 35 segundos de toma para calibración del oxímetro.

Vídeo con toma facial (Uma-rPPG)

A este dataset se lo denominó Uma-rPPG. Se realizó mediante la cámara RGB de un dispositivo móvil Motorola G8 Plus obteniendo videos con las siguientes características:

- Los videos fueron grabados con una frecuencia de 30 fps.
- La dimensión del video fue de 1080x1920.
- El formato del video obtenido fue mp4.

La toma de datos se llevó a cabo dentro de un espacio cerrado a lo largo de varios días hasta tomar las muestras necesarias con variaciones en la iluminación indirecta natural o artificial como únicas fuentes de luz.

Los sujetos se sentaron a una distancia de la cámara de 50 cm, sin presencia de otras personas en el cuadro del video (ver figura 3.1). Se tomaron los rostro completos de los sujetos, los cuales se encontraban centrados en el cuadro. Se sincronizaron previo al inicio con el oxímetro de pulso (ver sección 3.3.1 para detalles del oxímetro). Además, los sujetos debieron remover cualquier objeto o prenda que pudiese tapar su rostro. En caso de tener barba o anteojos, se los registró tal como fue indicado en la sección 3.3.1. Se les indicó a los sujetos que mirasen a la cámara y que sus caras completas fuesen vistas en el cuadro de video durante el tiempo de duración de la grabación, permaneciendo sentados y minimizando sus movimientos lo máximo posible.

Vídeo con toma en el dedo (Uma-dPPG)

Los videos se tomaron con la cámara trasera de un dispositivo móvil, modelo Google Pixel 4. Con esta cámara se tomaron los videos con las siguientes características:

- Los videos fueron grabados con una frecuencia de 30 fps.
- La dimensión del video fue de 1080x1920.
- El formato del video obtenido fue mp4.

Se consideró necesario un tercer celular para evitar que el LED del flash se caliente en demasía generando incomodidad o dolor a los sujetos. Este incluía las mismas características que el primer dispositivo. Para la captura del video se cumplieron las siguientes condiciones:

- Durante la captura del video los sujetos permanecieron en reposo, sentados en una silla. La linterna del dispositivo debió permanecer encendida durante toda la captura.
- Los sujetos cubrieron tanto la cámara trasera como la linterna del dispositivo móvil con su dedo índice de la mano derecha durante toda la captura. Aquellos que no cumplieron con dicho requisito fueron descartados (ver figura 3.1).
- Para asegurar la mayor estabilidad posible del video, durante la captura los sujetos apoyaron su mano derecha en una mesa mientras sujetaban el dispositivo.
- Se sincronizó previo al inicio con el oxímetro de pulso (ver sección 3.3.1).

A este dataset se lo denominó Uma-dPPG.

Uma-dPPG-extensión

En el caso de los videos del dedo, al tratarse de un dataset demasiado controlado en cuanto a las condiciones de captura de video, que no permitiría lograr robustez en los resultados, se realizó una ampliación del mismo para asemejar las condiciones a las del caso de uso real.

Para esto se tomaron 25 videos extra siguiendo la metodología descripta en la sección [3.3.1](#), pero capturándolos con distintos dispositivos móviles, y en distintas condiciones de iluminación. A este dataset se lo nombró Uma-dPPG-extensión.

Dataset de testeo

Además, se capturaron 40 videos del dedo de 10 segundos cada uno, 20 correspondientes a frecuencias cardíacas mayores a 100 lpm, y 20 correspondientes a frecuencias cardíacas menores a 100 lpm, para realizar el testeo del modelo. Estos videos fueron capturados con distintas condiciones de iluminación ambiental, y con distintos dispositivos móviles.

Para obtener este dataset se le pidió a 20 sujetos que capturen y envíen 2 videos de 12 segundos cada uno: uno en reposo, y uno habiendo hecho actividad física de manera que un oxímetro de pulso marque una frecuencia mayor a 100 lpm. Se les indicó que los videos debían ser capturados con la linterna de su dispositivo móvil prendida, y que debían cubrir completamente la cámara con su dedo. También se les indicó que en simultáneo midiesen su frecuencia cardíaca con un oxímetro de pulso y que la indiquen al enviar los videos para utilizar como etiqueta. Cada persona capturó los videos con su dispositivo móvil personal, y desde su hogar. De esta manera se intentó simular lo mejor posible las posibles condiciones de uso reales y con este dataset poder evaluar la robustez del modelo.

Estos 40 videos y sus respectivas etiquetas (frecuencia cardíaca medida con oxímetro de pulso) fueron utilizados como dataset de testeo para el desarrollo de modelos FPG de contacto.

Descripción de la población

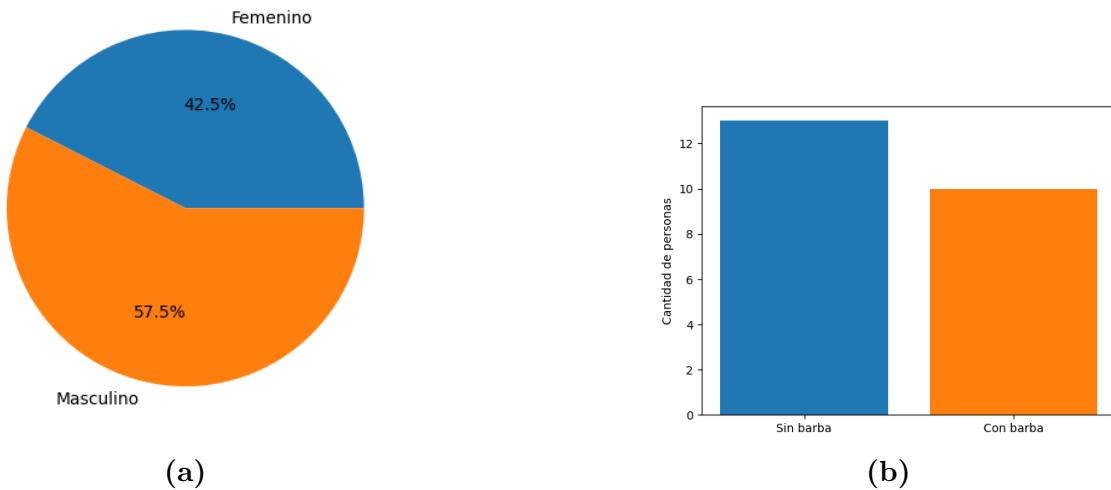


Figura 3.2: Características de la población que compone el dataset UMA-rPPG y UMA-dPPG:
(a) Sexo y (b) Barba que oculta parte del rostro.

Las muestras de los datasets UMA-rPPG y UMA-dPPG fueron recolectadas de 40 sujetos con edades entre los 20 y 40 años . De dichas personas, 3 poseía condiciones respiratorias crónicas (asma) y ningún sujeto declaró cardiopatías diagnosticadas. Dentro de la población, 23 eran de sexo masculino (ver figura 3.2a) de los cuales 10 poseían barba tupida (ver figura 3.2b). Esta característica fue considerada dado que la frecuencia cardíaca es estimada a partir de la variación de intensidad en la piel la cual se encuentra parcialmente cubierta en personas que contengan esta característica.

Métodos de almacenamiento

El servicio elegido para almacenar los datos fue Cloud Storage [86] de Google el cual permite almacenar en servidores remotos datos inmutables no estructurados denominados objetos. Estos objetos pueden ser archivos de cualquier formato, por ejemplo, videos y archivos csv. Los objetos se almacenan en contenedores llamados buckets. Todos los buckets están asociados con un proyecto que, a su vez, se puede agrupar en una organización, en este caso, de Üma Health S.A.

Dentro del bucket del equipo, se creó una carpeta asociada al proyecto (ver figura 3.3), la cual se divide en:

- Carpeta 'Etiquetas': dentro de la misma se almacenan los archivos con los datos tabulares o señales que serán utilizados como etiquetas. El nombre de cada archivo corresponde al dataset con el que se encuentra asociado. Las etiquetas están asociadas al video a través del mismo nombre del archivo que corresponde a un id único en todos los datasets. Estos datos fueron guardados en un archivo csv.

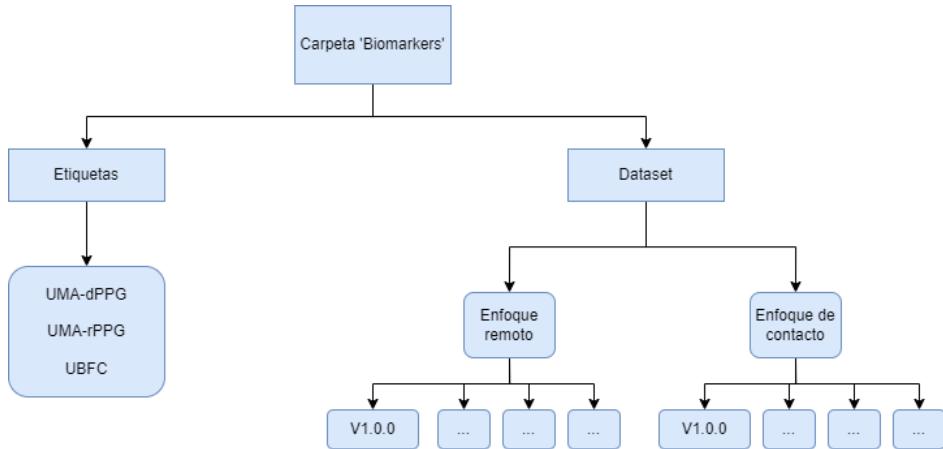


Figura 3.3: Estructura de carpetas dentro de Cloud storage para los datasets del proyecto.

- Carpeta 'Datasets': dentro de la carpeta se divide en los dos enfoques (remoto y de contacto). A su vez, en las carpetas se divide a partir del versionado de los datasets codificado a partir de 3 dígitos:
 - El primer dígito corresponde al dataset base a partir del cuál de le hicieron las modificaciones: UMA, UBFC.
 - El segundo dígito corresponde al largo de los clips del video. Hasta la entrega del informe, las posibilidades son el largo original (0), 300 frames (1) y 128 frames(2).
 - El tercer dígito corresponde a los procesamientos aplicados a los videos. El video original contiene la etiqueta 0. A medida que se le fueron realizando preprocesamientos a los datasets, se fue modificando dicho dígito y documentando el dígito asociado al procesamiento. Por ejemplo: al realizar una segmentación del área de interés, el tercer dígito se modificó a 1.

De esta forma, el dataset UMA-dPPG original con largo de 35 segundos le correspondió la carpeta V1.0.0 en la carpeta del enfoque de contacto.

Actualmente dichos datos no están abiertos al público pero se considera en un futuro su disponibilidad.

3.3.2. Dataset público

En el caso de los videos de la cara, al no poder conseguir resultados mejores únicamente con los datos generados, se buscó complementar con datasets públicos y abiertos disponibles en internet y reentrenar las redes con el mismo preprocesamiento aplicado al dataset propio. En esta búsqueda, se encontró el dataset UBFC-rPPG.

UBFC-rPPG

El dataset UBFC-rPPG [85] está compuesto por dos conjuntos de datos que se focalizan específicamente para el análisis de fotopletismografías remotas, formado por la Universidad Bourgogne Franche-Comté.

La base de datos fue creada utilizando una aplicación de C++ propia para la adquisición de videos con una webcam de bajo costo (Logitech C920 HD Pro) a 30 frames por segundo y una resolución de 640x480 píxeles en un formato RGB sin comprimir de 8-bit.

Un oxímetro de pulso transmisivo CMS50E fue utilizado para obtener la señal de onda de fotopletismografía, la cual se utilizó como el etiquetado de los videos. Durante la grabación de los videos, los sujetos se sentaron en frente de la cámara a aproximadamente un metro de distancia con la cara de forma visible. Todos los experimentos fueron conducidos en el interior de una habitación con una cantidad de luz solar e iluminación variada.

Los videos y los datos obtenidos de la muestra eran almacenados en una carpeta de Google Drive con acceso restringido otorgado a pedido. No se especificaban datos poblacionales del dataset.

Se tomaron 8 videos del primer dataset. En los mismos, a los participantes se les pidió permanecer quietos pero en varios de los videos hay presencia de movimiento de forma significativa, especialmente al principio de los mismos.

Para el segundo dataset que conforma UBFC-rPPG, los sujetos se sentaron aproximadamente a un metro de distancia de la cámara con la cara visible y se les pidió que jugaran un juego matemático con tiempo que apuntaba a aumentar la frecuencia cardíaca mientras simultáneamente emulaba un escenario de interacción humano-computadora normal. En este dataset se tomaron 42 videos.

3.4. Métodos aplicados a los videos obtenidos del dedo índice de los sujetos

Con los videos del dedo se entrenaron modelos de inteligencia artificial. Para esto se combinaron algoritmos de procesamiento y modelos de Deep Learning para obtener señales a partir de los videos. Luego con esas señales se entrenaron arquitecturas de redes neuronales para la predicción de frecuencia cardíaca.

Todos los procesamientos y modelos fueron programados en lenguaje Python. Para el armado de estructuras de redes neuronales y el entrenamiento de las mismas se utilizó la librería Keras, de TensorFlow. Para el procesamiento de los videos y datos se utilizaron las siguientes librerías: OpenCV, NumPy, Pandas y SciPy. Para el cálculo de resultados y métricas se utilizó la librería Scikit-learn.

3.4.1. Preparación de datasets

Se procesaron tanto el dataset Uma-dPPG como el Uma-dPPG-extensión. Estos datasets estaban compuesto por 87 videos (Uma-dPPG) y 25 videos (Uma-dPPG-extensión) de 35 segundos con una frecuencia de muestreo de 30 fps. Cada video se correspondía con 3 etiquetas de frecuencia cardíaca, una cada 10 segundos. Estos videos fueron recortados cada 10 segundos, descartando los primeros 5 segundos, para obtener los videos que se utilizaron para entrenar los modelos. La cantidad final de videos para entrenar fueron 261 (uma-dPPG) y 75 (Uma-dPPG-extensión)

3.4.2. Obtención de señal de FPG

Como se explicó anteriormente, la idea de una vFPG es utilizar una cámara de video como sensor para realizar un FPG. El video se captó con una frecuencia de 30Hz y las intensidades de cada canal fueron almacenadas en una escala de 0 a 255 [87]. Para generar una FPG a partir del video, se analizó la variación de intensidad de cada canal a través de la secuencia de imágenes.

En primer lugar, se seleccionó una región de interés (ROI, por sus siglas en inglés) en la imagen en donde se estimó que se encontraba la señal. Dado que el dedo de los sujetos en los videos obtenidos cubría la cámara en su totalidad, el cuadro completo de la imagen contenía información. No obstante, se consideró que las fuentes de luz de diferentes dispositivos de captura de video podrían estar localizadas en distintos puntos y, en consecuencia, los límites de la imagen podrían variar en su intensidad entre dispositivos, lo que generaría ruido en el video (ver figura 3.4). Por esta razón, se descartó un 10 % de cada límite de cada imagen [88] (ver figura 3.5), tomando un rectángulo central de cada imagen como ROI cuya iluminación suele ser más uniforme en todos los dispositivos.

A continuación, para cada imagen del video, se calculó por canal el promedio de los valores de los píxeles que se encuentran dentro de la ROI, y se almacenaron estos promedios secuencialmente en un vector. De esta manera, se obtuvieron tres señales (una por cada canal) con la misma frecuencia de muestreo que el video original, las cuales se consideraron como candidatas a FPG (ver figura 3.5).

3.4.3. Entrenamiento de un primer modelo de predicción de frecuencia cardíaca

En primer lugar, con las señales obtenidas a partir de los videos del dataset Uma-dPPG y sus respectivas etiquetas de frecuencia cardíaca, se entrenaron 3 redes neuronales convolucionales 1D.

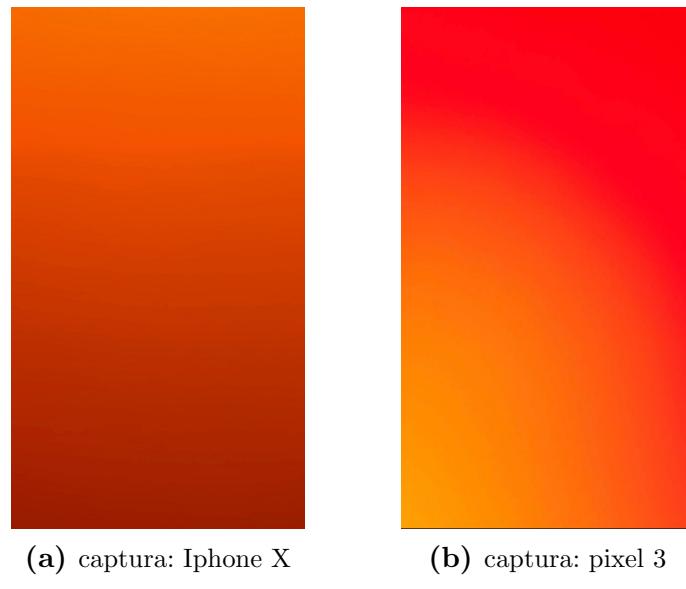


Figura 3.4: Imágenes de videos del dedo cubriendo la cámara tomados con dos dispositivos distintos. En el cuadro del video (a), la mayor iluminación se encuentra en la porción superior, mientras que en el video (b) la porción iluminada se encuentra en la esquina inferior izquierda.

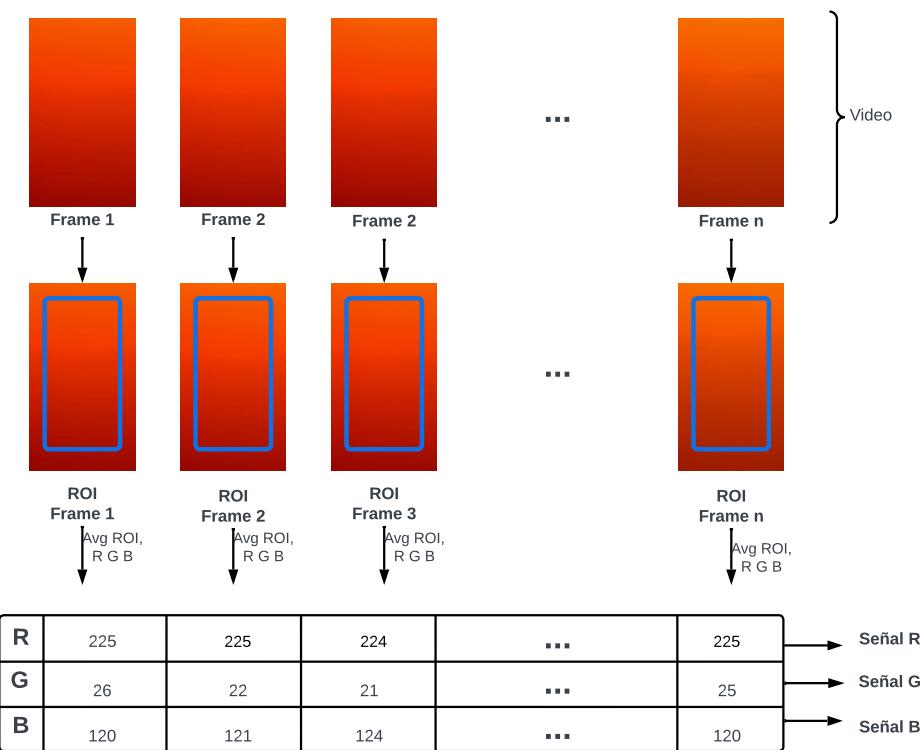


Figura 3.5: Proceso de obtención de tres señales candidatas a FPG a partir de un video.

Las redes neuronales convolucionales 1D son un tipo de red neuronal que se utiliza en el aprendizaje profundo. Como su nombre lo indica, las redes neuronales convolucionales 1D se basan en la idea de la convolución, una técnica matemática que permite extraer características de una señal en una dimensión (como una secuencia de audio o una serie temporal de datos).

Las redes neuronales convolucionales 1D se utilizan principalmente en tareas de procesamiento de señales, como el reconocimiento de habla, la detección de patrones en series temporales o la generación de música. Estas redes son muy efectivas para extraer características relevantes de señales de una sola dimensión, lo que las hace ideales para aplicaciones en las que se manejen datos secuenciales.

Es por esto que resultó útil utilizar este tipo de red neuronal para extraer las características principales de las señales de FPG y predecir la frecuencia cardíaca. Esto se debe a que se desea encontrar las características temporales (única dimensión) de las señales de entrenamiento (posición temporal de los picos, y distancia temporal entre los mismos), y con estas características obtener la frecuencia cardíaca.

Para el entrenamiento de estos modelos, las señales obtenidas de los 361 videos fueron alimentadas a redes neuronales con la arquitectura que se muestra en la tabla 3.2. Esta red toma tres señales como entrada (una correspondiente a cada canal). Sus capas ocultas consisten en 3 bloques formados por una capa convolucional 1D con 64 filtros de tamaño de kernel 3 y función de activación ReLu, seguida por una capa de Max Pooling. Finalmente el output, es decir, la frecuencia cardíaca, es generado por una capa densa de regresión.

Nº Bloque	Capa	Formato de salida	Cantidad de parámetros
1	Conv1D	(None, 240, 64)	7744
1	MaxPooling1D	(None, 120, 64)	0
2	Conv1D	(None, 91, 64)	122944
2	MaxPooling1D	(None, 45, 64)	0
3	Conv1D	(None, 16, 64)	122944
3	MaxPooling1D	(None, 8, 64)	0
-	Flatten	(None, 512)	0
-	Dense	(None, 1)	513

Tabla 3.2: Estructura de la red convolucional 1D correspondiente a los modelos 1DCNNv1.x

Se entrenaron 6 modelos variando el learning rate y la cantidad de epochs de entrenamiento. Los parámetros utilizados en cada modelo se muestran en la tabla 3.3

Modelo	Learning Rate	Cantidad de epochs
1DCNNv1.1	1	50
1DCNNv1.2	0.1	100
1DCNNv1.3	0.01	150
1DCNNv1.4	0.008	200
1DCNNv1.5	0.005	200
1DCNNv1.6	0.001	250

Tabla 3.3: Parametros utilizados para el entrenamiento de cada red neuronal convolucional 1d

3.4.4. Análisis de señales de FPG

Debido a la poca robustez de los primeros modelos y su incapacidad de generalizar frente a videos con una captura diferente a la de entrenamiento (ver resultados en sección 4.1.1), se realizó un análisis de las señales. Este análisis tuvo la finalidad de identificar puntos débiles en los primeros modelos entrenados y entrenar nuevos modelos que puedan lidiar con dichos puntos débiles.

De los 3 canales de color disponibles en el video, la señal obtenida del canal rojo fue propensa a representar de mejor manera la frecuencia cardíaca[89][90] [91]. De todas maneras, para corroborar estos resultados en el dataset obtenido e identificar posibles preprocesamientos a realizar en las señales, se analizaron las señales obtenidas con cada uno de los 3 canales para una selección de videos (de los dataset propios y el dataset de validacion). En la figura 3.6 se muestran a modo de representación las señales correspondientes a cada canal obtenidas a partir de 3 videos del dataset Uma-rPPG, mientras que, en la figura 3.7 se muestran las señales obtenidas a partir de 3 videos del dataset Uma-rPPG-extensión.

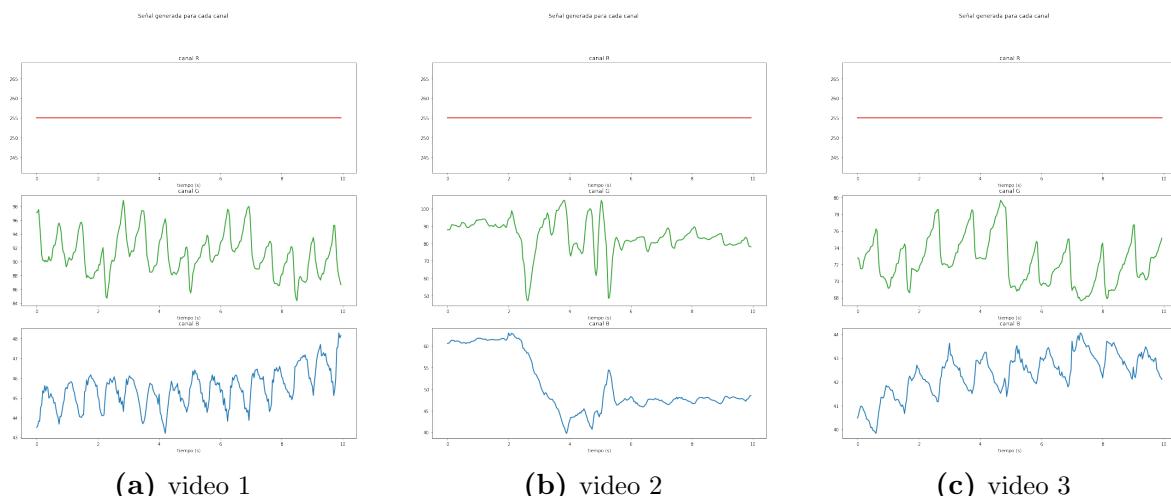


Figura 3.6: Señales obtenidas con 3 videos pertenecientes al dataset Uma-rPPG

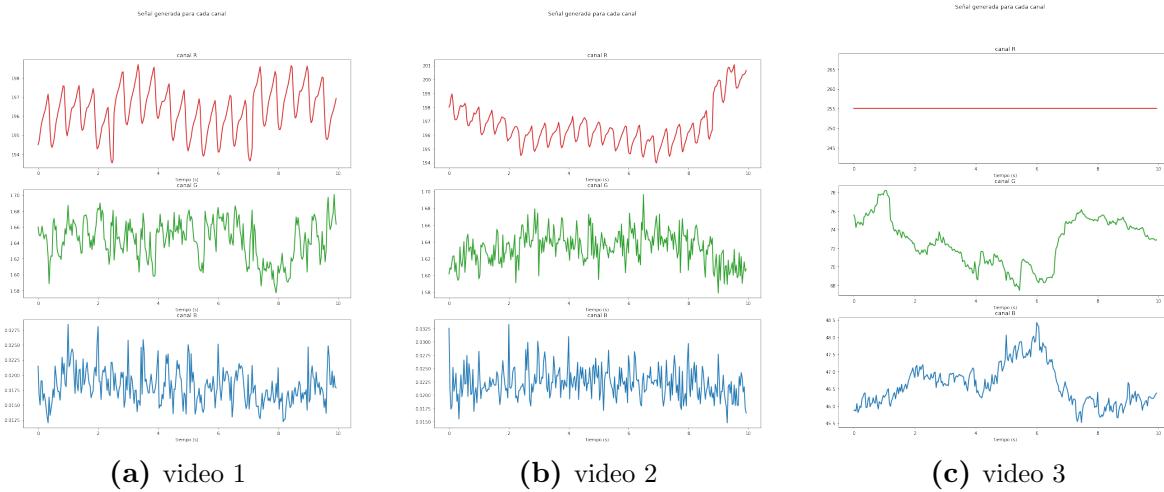


Figura 3.7: señales obtenidas con tres videos pertenecientes al dataset Uma-rPPG-extensión

Analizando las imágenes de las figuras 3.6 y 3.7 se determinaron los siguientes puntos:

- Se observó que en el dataset Uma-dPPG, contrario a lo sugerido en la bibliografía[89–91], la señal obtenida con el canal rojo se encontraba saturada completamente en todos los casos analizados, mientras que la señal que mejor representaba el FPG era la señal correspondiente al canal verde o azul.
 - Se identificó que la señal de FPG se podía encontrar en cualquiera de los tres canales en el dataset Uma-dPPG-extensión, y variaba según el dispositivo de captura. Esto fue atribuido a que cada dispositivo contiene un procesamiento de video propio y, por lo tanto, genera los colores de las imágenes de diferentes maneras.
 - Se observó que en algunos casos, tanto en el dataset Uma-dPPG como en Uma-dPPG-extensión, la señal generada en todos los canales contenía ruido, hasta un punto que no se podía diferenciar en ella ninguna señal de FPG. Esto fue atribuido a movimientos bruscos o cambios repentinos de luz durante la captura de video. Cuando sucedía alguna de estas situaciones, el dispositivo ajustaba los parámetros de iluminación de video, y también así los colores de cada canal. Este ajuste demoraba unos segundos, lo que provocaba que la pérdida de la señal de FPG en parte del video.

A partir de este análisis, se dedujo que era necesario verificar la presencia de la señal de FPG en al menos uno de los tres canales del video y, ante un video que contuviese la señal, determinar el canal en el que se encontrase la señal de FPG con menor ruido y mejor resolución. Con esta propósito, se entrenó un modelo clasificador de señales.

Incorporar esta constatación al entrenamiento del modelo de predicción de la frecuencia cardíaca y, posteriormente, al flujo final del algoritmo presentó múltiples beneficios. A la hora de conformar el dataset de entrenamiento, este se compuso por señales de buena calidad lo cual generó un modelo con menor error y, por ende, más confiable. Así mismo, la elección del canal óptimo para extraer la señal generó un modelo más robusto capaz de generalizar frente a videos provenientes de distintos dispositivos.

3.4.5. Entrenamiento de una red convolucional 2D clasificadora de señales

Para determinar si un video contiene o no una señal de FPG, y seleccionar en qué canal esa señal es más fuerte, se entrenó una red neuronal convolucional 2D. Esta red se entrenó para clasificar imágenes que contienen un gráfico de la señal obtenida con un canal de video en dos categorías: señal de FPG y señal no correspondiente a FPG.

Se decidió entrenar este tipo de modelo por diversas características que permiten que se cumplan las funciones deseadas. En primer lugar, hay mayor cantidad de investigaciones, y con mejores resultados, de modelos clasificadores de imágenes (2D), que clasificadores de señales (1D), por lo que resulta más simple y conocido tomar los gráficos de las señales obtenidas con imágenes para cumplir esta tarea. A esto se le suma el acceso a modelos preentrenados como VGG y ResNet los cuales facilitan la obtención de resultados con buen desempeño en clasificación 2D a través de técnicas de Transfer Learning. Estos modelos resultan favorables para la selección de la mejor señal debido a su capacidad para producir probabilidades de pertenencia a una clase mediante el uso de una función softmax en su última capa. La función softmax transforma las salidas del modelo en distribuciones de probabilidad, asignando un valor entre 0 y 1 a cada clase. Esta probabilidad se interpreta como un puntaje de confianza para cada clasificación o para cada señal analizada. Por lo tanto, se puede utilizar este enfoque para los gráficos de los tres canales y seleccionar el gráfico que tenga la mejor señal. Es decir, incluso si se detecta una señal en los tres canales, se puede identificar en qué canal se encuentra la mejor señal en función del puntaje asociado. Esto se determina mediante la evaluación de las probabilidades calculadas por el clasificador para cada canal. El canal que tenga la mayor probabilidad de contener una señal se considera como el que presenta la mejor señal.

Para entrenar el modelo clasificador, en primer lugar se generó el dataset, con las imágenes a clasificar y sus respectivas clases. Para esto se graficó para cada video del dataset Uma-dPPG y Uma-dPPG-extensión la intensidad de cada canal en función del tiempo. Cada gráfico se guardó como imagen jpg. De esta manera, por cada video se obtuvieron 3 gráficos de señales (uno por cada canal), algunas correspondientes a una señal de FPG, y algunas no.

A continuación, se llevó a cabo la clasificación manual de las imágenes para generar las dos clases requeridas. Se examinó cada imagen para determinar si el gráfico representado correspondía o no a una señal de FPG, ya que, como se mostró anteriormente, esto era algo identificable a simple vista. Se clasificaron 515 imágenes, de las cuales 303 contenían una señal de FPG y 212 no contenían una señal de FPG.

Posteriormente, se procedió a entrenar dos CNN 2D, utilizando la metodología de Transfer Learning y fine tuning. Las arquitecturas utilizadas fueron VGG y ResNet, ambas previamente entrenadas con el conjunto de datos de ImageNet.

La arquitectura VGG (Visual Geometry Group) es una red neuronal convolucional 2D conocida por su estructura profunda y homogénea. Se caracteriza por su simplicidad y por emplear convoluciones de tamaño pequeño (3×3) con capas de max pooling intercaladas. Esta arquitectura se ve ilustrada en la figura 3.8. El diseño de la red se basa en apilar múltiples capas convolucionales y totalmente conectadas, con un total de 16. La arquitectura VGG logra una gran capacidad de aprendizaje al tener un número significativo de capas convolucionales, lo cual permite aprender características cada vez más complejas a medida que se profundiza en la red. Esta arquitectura ha demostrado buen rendimiento en la clasificación de imágenes, especialmente en tareas de reconocimiento visual.

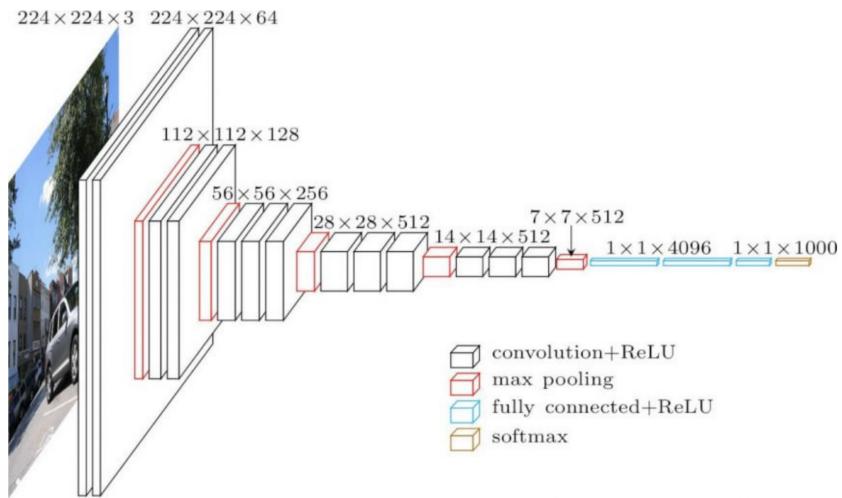


Figura 3.8: Arquitectura de red VGG.

Por otro lado, la arquitectura ResNet (Residual Network) se basa en la idea de bloques residuales. Introduce conexiones residuales que saltan sobre una o varias capas convolucionales, permitiendo que la información se transmita de forma directa a capas posteriores. Estas conexiones residuales ayudan a mitigar el problema del desvanecimiento del gradiente durante el entrenamiento de redes neuronales profundas. La clave de esta arquitectura es la utilización de los denominados 'skip connections' que saltan una o más capas convolucionales. De esta manera, la información original se conserva y se suma a la información aprendida por las capas subsiguientes. Esto permite el

entrenamiento de redes neuronales extremadamente profundas, con más de 100 capas, sin sufrir una degradación significativa en el rendimiento. Este funcionamiento se ve ilustrado en comparación con el comportamiento de la arquitectura VGG en la figura 3.9.

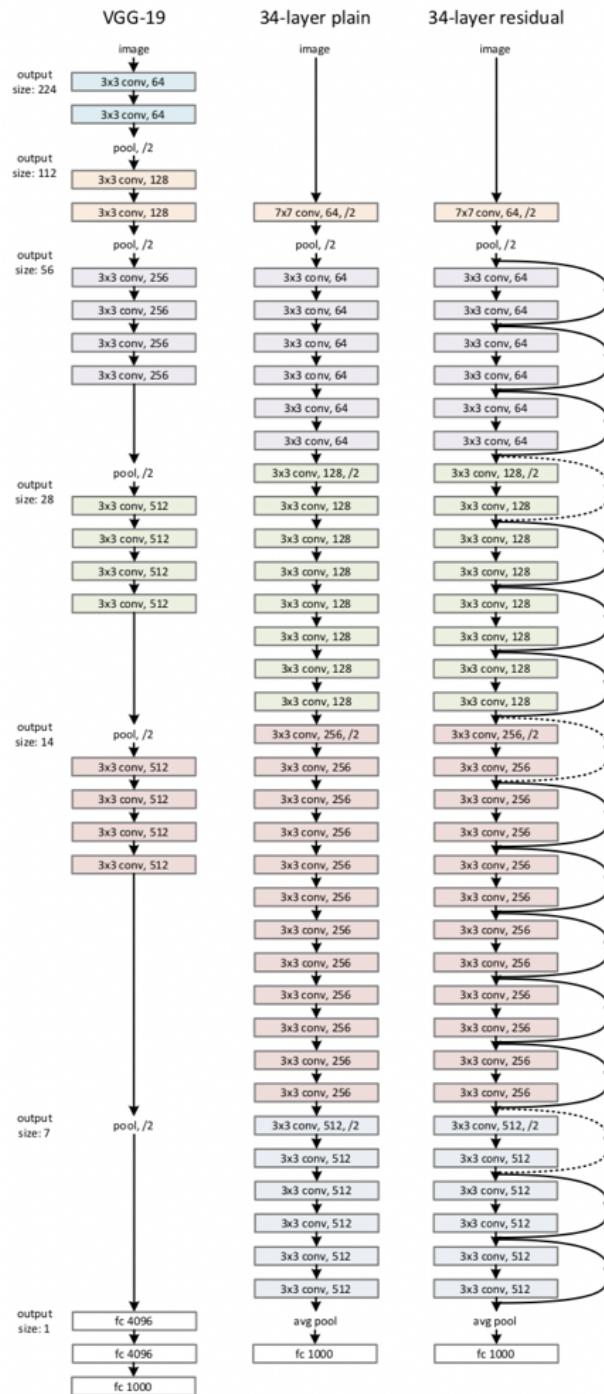


Figura 3.9: Arquitectura de red ResNet.

Para el entrenamiento del modelo, se eliminó la última capa clasificadora de cada red y se sustituyó por una nueva capa densa que clasificara en dos clases (señal de FPG y no señal de FPG). Todas las capas, excepto las últimas cuatro, fueron congeladas

durante el entrenamiento, es decir, sus pesos se mantuvieron fijos. Solo las últimas cuatro capas se entrenaron con el objetivo de adaptar el modelo a la tarea específica de clasificación de señales de FPG. Se entrenó cada modelo durante 40 epochs con un learning rate de 0.01. Luego, para mejorar el ajuste de los modelos, se descongelaron todas las capas y se entrenó todo el modelo durante 10 epochs con un learning rate de 0.0001.

Para seleccionar el mejor modelo se midieron y se compararon la exactitud, precisión, exhaustividad y F1-score de ambos modelos.

Todo este procesamiento se utilizó para validar que en el video se pudiese encontrar una señal utilizable y seleccionar en qué canal se encuentra la señal más adecuada en cada caso. Este proceso se aplicó tanto para la generación de señales para el dataset de entrenamiento de un modelo predictor de frecuencia cardíaca, como en el flujo final para la predicción de frecuencia a partir de un video (ver figura 3.10).

Tal como se muestra en la figura 3.10, a partir de un video se obtuvieron tres señales candidatas a FPG, una por cada canal. Cada señal fue graficada, y la imagen de cada gráfico fue clasificada por el clasificador. Como resultado se obtuvieron tres 'puntajes', los tres outputs del modelo, que determinaron que tan probable era que cada señal fuese de FPG. Se compararon los tres puntajes, y se seleccionó el más alto como el perteneciente a la mejor señal de FPG. Si este puntaje era menor a 0.4 se asumía que el video no contenía una señal de FPG. El video se descartaba, en el caso de estar preparando el dataset de entrenamiento, o se pedía una nueva captura en el caso de estar realizando una predicción. Si el puntaje era mayor a 0.4 se devolvía como output del flujo la señal que arrojaba este puntaje, es decir, la mejor señal de FPG obtenida con ese video.

3.4.6. Preprocesamiento de señales

A continuación se utilizó el modelo anterior para determinar las señales a utilizar en el modelo de frecuencia cardíaca. De 435 videos de 10s se determinó que 382 de ellos contenían señales utilizables, por lo que se descartaron 53 videos. El modelo también permitió determinar qué señal utilizar en cada caso para el entrenamiento del modelo.

A las señales utilizadas luego se las normalizó y se les aplicó un filtro pasa bandas Butterworth de orden 2, en el rango de los 0.75 a los 3 hz, ya que es en este rango que se encuentra la frecuencia cardíaca humana (45 a 180 lpm). Este filtro ayuda a eliminar el ruido proveniente de señales ambientales que no corresponden a la frecuencia cardíaca [91–93]. (ver figura 3.11)

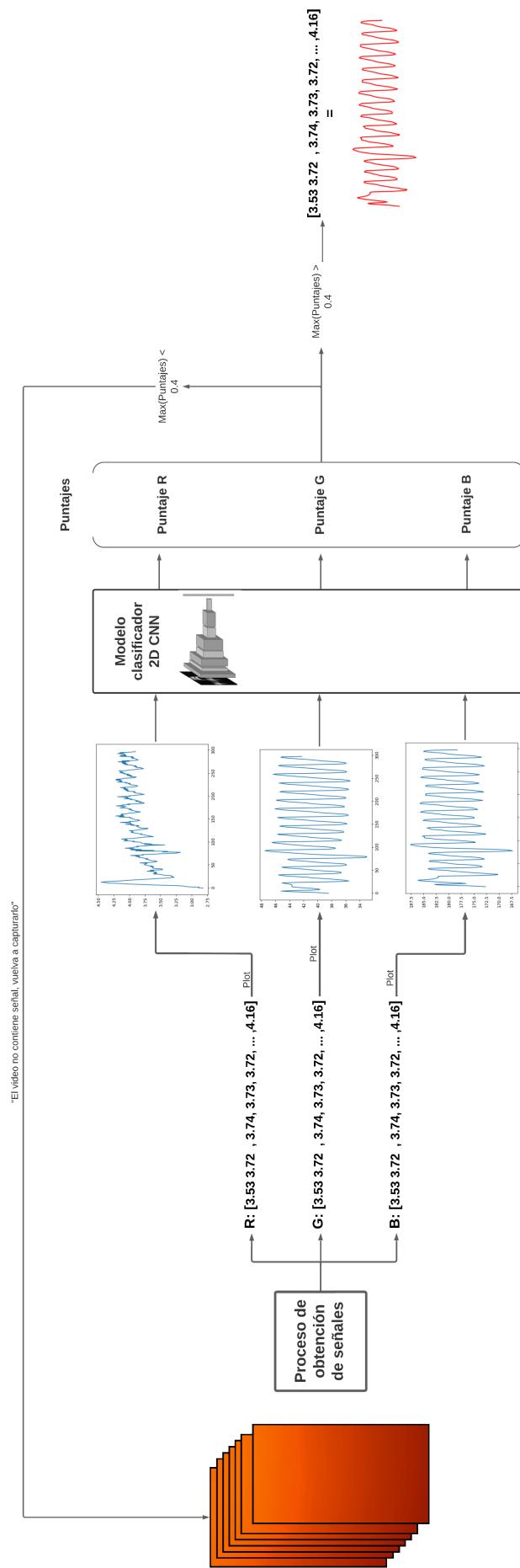


Figura 3.10: Proceso de selección de mejor señal de FPG correspondiente a un video utilizando un modelo clasificador de señales

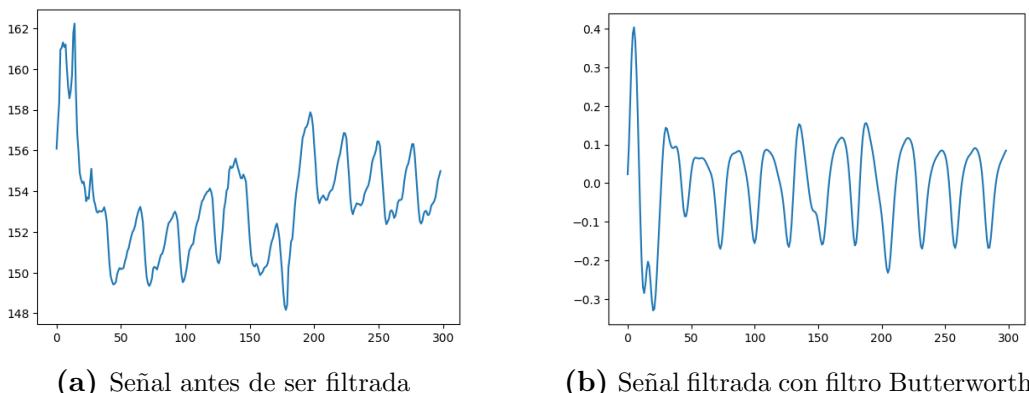


Figura 3.11: Efecto de filtro Butterworth de orden 2 en señal de video FPG

3.4.7. Entrenamiento de modelos de predicción de frecuencia cardíaca

Con las 362 señales obtenidas con el procesamiento anterior y sus etiquetas de frecuencia correspondientes, se entrenó una red convolucional 1D para predecir la frecuencia cardíaca a partir de una señal. Se entrenaron 6 modelos distintos, todos ellos con arquitecturas convolucionales 1D, pero variando ligeramente las arquitecturas, preprocesamientos y/o parámetros de entrenamiento.

Las arquitecturas de los modelos entrenados consisten en: una serie de bloques formados por una capa convolucional y una capa de pooling, seguidos por una flatten, y una capa densa, que comprime los valores obtenidos en un único output (la frecuencia cardíaca). Las capas convolucionales tienen función de activación ReLu. Las capas de pooling eran MaxPooling, con tamaño de pool 2. En la tabla 3.4 se muestra la arquitectura correspondiente al modelo v4. Las demás arquitecturas se muestran en el Anexo.

Nº Bloque	Capa	Formato de salida	Cantidad de parámetros
1	Conv1D	(None, 293, 30)	240
1	MaxPooling1D	(None, 146, 30)	0
2	Conv1D	(None, 140, 30)	6330
2	MaxPooling1D	(None, 70, 30)	0
3	Conv1D	(None, 64, 30)	6330
3	MaxPooling1D	(None, 32, 30)	0
4	Conv1D	(None, 26, 30)	6330
4	MaxPooling1D	(None, 13, 30)	0
5	Conv1D	(None, 7, 30)	6330
5	MaxPooling1D	(None, 3, 30)	0
-	Flatten	(None, 90)	0
-	Dense	(None, 1)	91

Tabla 3.4: Estructura de la red convolucional 1D correspondiente al modelo v4

Modelo	Cantidad de bloques	Cantidad de filtros por capa	Tamaño de kernel	Otras características de arquitecturas	Preprocesamiento de señales	Learning rate: cantidad de epochs
1DCNNv2.1	3	64, 64, 64	60, 30, 30	-	Normalización	0,01:100 0,001:200
1DCNNv2.2	5	10, 10, 10 10 10	3, 3, 3, 3, 3	-	Normalización, Filtros	0,01:50 0,001:200 0,0001:200
1DCNNv2.3	5	30, 30, 30, 30, 30	5, 5, 5, 5, 5	-	Normalización, Filtros	0,01:50 0,001:200
1DCNNv2.4	5	30, 30, 30, 30, 30	7, 7, 7, 7, 7	-	Normalización, Filtros	0,01:50 0,001:200
1DCNNv2.5	5	50, 50, 50, 50, 50	7, 7, 7, 7, 7	Dropout 0.25	Normalización, Filtros	0,01:50 0,001:200 0,0001:200
1DCNNv2.6	5	30, 30, 30, 30, 30	5, 5, 5, 5, 5	-	Normalización	0,01:100 0,001:200

Tabla 3.5: Hiperparámetros de arquitectura y entrenamiento de cada modelo

En las distintas arquitecturas se variaron el tamaño del kernel, la cantidad de filtros por capa convolucional y la cantidad de bloques de la red. En el entrenamiento de cada modelo se variaron el learning rate, y la cantidad de epochs. Estas pequeñas variaciones se realizaron para detectar el mejor modelo. Las características de cada modelo se muestran en la tabla 3.5.

Los modelos se entrenaron utilizando las señales de FPG como input y los valores de frecuencia cardíaca como output. Al tratarse de un modelo de regresión, con outputs continuos, se utilizó el MAE de las frecuencias cardíacas como función de pérdida (ecuación 2.21). Esto se debe a que lo que se busca con el modelo no es que la mayor cantidad de predicciones sean exactamente igual a las frecuencias cardíacas originales, sino que la totalidad de las predicciones se asemejen lo máximo posible a las frecuencias cardíacas originales.

Para el entrenamiento se utilizó el método de hold out validation, es decir, de las 382 disponibles, se utilizó el 0.9 (344 señales) para el entrenamiento de la red, y se reservó el 0.1 (38 señales) para la validación del modelo y detectar el sobreentrenamiento.

3.4.8. Métodos de evaluación

En primer lugar se midió la exactitud del modelo clasificador de señales de FPG. También se calculó, para el dataset de entrenamiento y de testeo, la proporción de videos que quedarían descartados por este clasificador.

Para seleccionar la mejor arquitectura del modelo se comparó el MAE obtenido en cada caso para la partición del dataset de validación.

Luego, para evaluar el desempeño del mejor modelo y su capacidad de generalización frente a diferentes condiciones, se calculó el MAE, el RMSE y el R^2 para el dataset de testeo. También se calcularon estas métricas para el subgrupo de frecuencias cardíacas normales (<100 bpm) y el de frecuencias taquicárdicas (≥ 100 bpm)

3.5. Métodos aplicados a los videos obtenidos de la cara de los sujetos

El siguiente modelo generado fue el modelo predictor de frecuencia cardíaca a partir de un video de la cara. El principal desafío fue la normalización de los videos ya que se buscó que los videos pudiesen provenir de cualquier tipo de dispositivo siempre y cuando la frecuencia de muestreo de la cámara fuese igual o mayor que 30 fps. Con esto en mente, el video pasa por una serie de modificaciones hasta que es ajustado a las características de los datos de entrada de la red.

Las etapas de preprocesamiento (ver figura 3.12) fueron dos:

- Preparación del dataset, en la cual se ajusta la frecuencia de muestreo del video y el largo del mismo.
- Selección del área de interés, en la cual se realizó la segmentación de la cara en la orientación y tamaño adecuados para ser ingresado a la red.

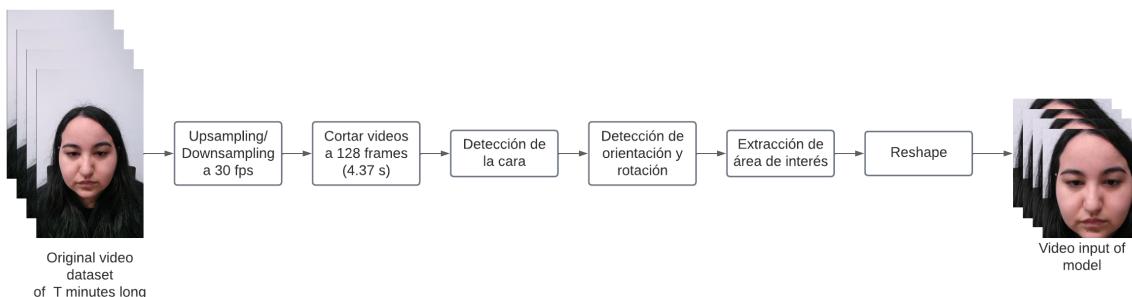


Figura 3.12: Flujo de preprocesamiento de los videos.

Una vez preparado el dataset, se entrenaron los distintos modelos seleccionados hasta obtener el mejor modelo posible. Con dicho modelo, se construyó el algoritmo para generar las predicciones de frecuencias cardíacas a partir de videos nuevos.

Para el desarrollo del modelo se utilizó la librería de Tensorflow programando en Python en el entorno de Google Colaboratory. Para las distintas etapas del flujo y manejo de los videos se utilizaron Opencv, Mediapipe, Ffmpeg y Heartpy.

3.5.1. Preparación de los datasets

Dataset UMA-rPPG

Inicialmente solo los videos del dataset propio, UMA-rPPG, fueron utilizados como entrenamiento y validación del modelo. El dataset estaba compuesto por 87 videos de

35 segundos con una frecuencia de muestreo de 30 fps. Estos videos fueron recortados cada 10 segundos, descartando los primeros 5 segundos, para obtener los datos que se utilizaron para entrenar el modelo. La cantidad final de videos para entrenar fueron 252 en 3 experiencias (reposo, apnea y tras ejercicio físico) de 261, dado que se descartaron algunos cuyo preprocesado no fue exitoso dadas condiciones de movimiento, cara cubiertas o fallas en los pasos siguientes.

UBFC

Como fue mencionado anteriormente, para aumentar la cantidad de datos de entrenamiento se utilizó la base UBFC-rPPG, la cual posee videos etiquetados con señales de FPG.

El primer paso de dicho preprocesamiento fue un re-muestreo de los videos, como medida preventiva frente a videos que contengan otro tipo de frecuencia de muestreo. Si bien esto implicó una pérdida de datos, la mayoría de los dispositivos utilizan una frecuencia de 30 fps y, al tener una cantidad limitada de frames dentro de un video, hay un menor costo computacional para los pasos siguientes en comparación a videos de mayor frames por segundo. Este paso también fue aplicado en el flujo final del algoritmo. Este evita un cálculo erróneo de la frecuencia cardíaca frente a malas configuraciones de la cámara. Dado que un video es una secuencia de imágenes capturadas a una frecuencia determinada y que el ritmo cardíaco depende de las características temporales de una señal, si al modelo se le entrega un video con distinta frecuencia de 30 fps, este puede distorsionar el resultado.

El siguiente paso es el recorte de los videos. En un principio, se seleccionó un largo de 300 frames. A este dataset (206 videos) se lo utilizó como dataset de testeo a aquellos modelos que fueron entrenados por UMA-rPPG.

Posteriormente, al tomarlo como dataset de entrenamiento, se seleccionó un largo de 128 frames para la duración del video, ya que es el largo que presentó mejores resultados en la bibliografía [32]. A partir de un solo video, se pudieron obtener más de un elemento para el dataset con este recorte lo que resultó en un aumento de datos. En total se obtuvieron 610 videos finales de 4.27 segundos cada uno para entrenar y validar el modelo.

3.5.2. Selección del área de interés

Dentro del cuadro de video, no toda la imagen posee información útil para la detección del FPG. Es por esto que el siguiente paso de preprocesamiento del video constó de seleccionar el ROI. Se realizaron pruebas con diversos métodos para el desarrollo de esta herramienta, entre los cuales se eligieron el algoritmo de Viola Jones con seguimiento de movimiento, y el algoritmo de Mediapipe.

Viola-Jones y seguimiento de movimiento

El primer algoritmo utilizado fue el detector de rostro Viola-Jones [12, 94] implementado con la librería de OpenCV. Los beneficios de implementar este algoritmo incluyen un ratio de detección alto, bajo costo computacional, bajos casos de falsos positivos y aplicable en tiempo real.

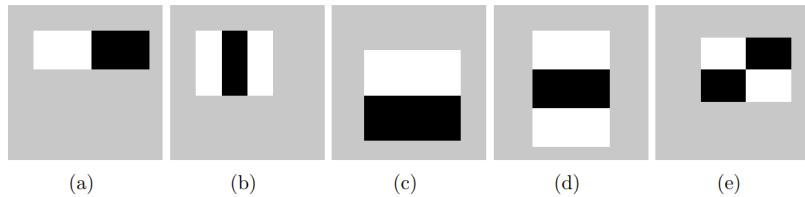


Figura 3.13: Patrones Haar para detección de rostros [12].

El algoritmo se basa en una serie de clasificadores débiles denominados Haar-like features que se pueden calcular eficientemente a partir de una imagen integral. Estas features son rasgos o características que se buscan en la imagen y para poder hallar estas características, la imagen primero se divide en una grilla de rectángulos. Las features se definen por la imagen y los patrones Haar (ver figura 3.13) y, consisten en la diferencia de intensidades luminosas entre regiones rectangulares adyacentes (ver figura 3.14).



Figura 3.14: Haar-like features [13].

La suma de los píxeles en un subset rectangular de la grilla puede ser calculada de manera muy eficiente empleando una representación intermedia denominada imagen integral. La imagen integral en el punto (x, y) contiene la suma de todos los píxeles que están arriba y hacia la izquierda de ese punto en la imagen original. De esta forma, el valor del pixel en la posición (x, y) puede obtenerse como

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \quad (3.1)$$

donde $ii(x,y)$ es el valor de un pixel en la posición (x, y) de la imagen integral, y $i(x, y)$ el valor de un pixel en la posición (x, y) de la imagen original.

El clasificador Haar Cascade se utiliza para detectar las caras en la imagen. El algoritmo aplica una búsqueda por ventaneo en las regiones para detectar las caras

con cada patrón generando un clasificador débil. A partir de estos, se construye un clasificador fuerte como la combinación lineal de todos los clasificadores débiles. Si encuentra suficientes coincidencias entre los patrones y la imagen, se considera que hay un rostro. A pesar de la velocidad y eficacia del Haar Cascade para detectar caras, este se debió realizar frame a frame en un video. En esta circunstancia, no siempre se desempeñó bien frente a movimiento del sujeto en el video y, en comparación a otras soluciones, resultó lento en tiempo real.

Para solucionar las fallas de detección frente a movimientos, se agregó un algoritmo de seguimiento de movimiento Lucas Kanade [95]. El algoritmo Lucas Kanade se basa en la asunción que los píxeles vecinos se moverán de forma similar y que, el brillo y color de un píxel no cambiará abruptamente de un frame a otro. El flujo óptico es un patrón de movimiento aparente de los objetos de la imagen entre dos fotogramas consecutivos causados por el movimiento del objeto o la cámara. Se define matemáticamente como un campo vectorial 2D, donde cada elemento es un vector de desplazamiento que muestra el movimiento de los puntos desde el primer frame al segundo. OpenCV provee este algoritmo mediante una única función. En el primer frame se detectan los puntos de la detección de rostro. Esos puntos serán trackeados con el flujo óptico usando Lucas Kanade el cual retorna los siguientes puntos y un puntaje de certeza de 1 a 0 (si el punto es encontrado o no, respectivamente). Este proceso se realiza de forma iterativa sobre todos los frames del video con dichos puntos.

Sin embargo, no solo incrementaba el tiempo de procesamiento, sino que el algoritmo fallaba frente a grandes movimientos de frame a frame. Por ejemplo: sacudir la cabeza o espasmos. Otra falla posible fue frente a movimientos que generasen la desaparición del punto en el cuadro, como voltear la cabeza. Si bien el algoritmo intentaba seguir el movimiento y seleccionar un punto, el cuadro original que segmentaba el rostro sufría deformaciones o los puntos elegidos eran los erróneos.

Mediapipe

Mediapipe [96] es un proyecto de código abierto de Google que ofrece acceso a múltiples modelos basados en Machine Learning y permite construir pipelines que realicen inferencias de acuerdo a datos como imágenes, videos o audio. Dentro de sus modelos, se encuentra el modelo de detección de rostros, Mediapipe Face Detection [97]. Este algoritmo es un modelo de detección facial rápido con 6 landmarks, o puntos de referencia, capaz de detectar múltiples rostros al mismo tiempo. Se basa en BlazeFace, un detector liviano y con buen desempeño, adaptado para la inferencia de GPU móvil. Su alto rendimiento permite que se aplique a tiempo real con una detección del área de interés precisa para realizar la segmentación de la cara en el cuadro del video.

En comparación a otros algoritmos actuales de detección de rostros, Mediapipe

presenta alta velocidad de procesamiento, especialmente frente a Haar cascade (225.34 frames por segundo contra 19.95, respectivamente)[13]. Ya que el objetivo del desarrollo es utilizarlo en la aplicación de Uma Health, la velocidad de procesamiento de este algoritmo permite tener el menor tiempo de ejecución posible.

Por estas consideraciones, se decidió incluir el algoritmo de Mediapipe Face Detection como algoritmo de detección de rostros en la versión final del algoritmo. La función de detección de rostros generada permite asegurarse que al modelo se le entreguen videos con un rostro en él. Si el algoritmo no detecta una cara, la predicción no se realiza. De realizarse, el algoritmo no sería confiable.

Una vez que un rostro es detectado, el algoritmo procede a analizar la orientación de la misma, considerando que el usuario podría filmar un video en cualquier orientación o en ciertas circunstancias un dispositivo puede rotar la orientación del video. Dado que los modelos se entrenaron con videos orientados en la misma dirección, se creó una función que detecta la orientación de la cara y rota el video de forma acorde para que quede orientado de la forma que debería. Una vez rotado el vídeo, el algoritmo procede a segmentar la cara detectada. El nuevo frame segmentado se le modifica su ancho y largo a 128px y se lo guarda en un nuevo video para entrenar los modelos.

3.5.3. Magnificación de la señal

El objetivo de la magnificación del video es aumentar la diferencia de los cambios en color que no son visibles para el ojo humano entre cada frame. Los enfoques logrados de la magnificación euleriana [98] no requieren mucho costo computacional y han demostrado ser particularmente efectivos para visualizar movimientos repetitivos, en los cuales se amplifican las variaciones en un pixel determinado.

A partir del video, la magnificación euleriana aísla los cambios de interés al aplicarle un filtro pasa-bajos espacial y un filtro pasa-bandas temporal. Aplicar el filtro espacial es equivalente a realizar un promedio espacial de cada frame del video, lo cual sirve para reducir el nivel de ruido y mejorar la relación señal-ruido. El segundo filtro es utilizado para aislar las frecuencias cercanas a la frecuencia cardíaca. Los cambios son multiplicados por un factor constante y recombinados con el video de entrada para crear un nuevo video en el cual estos cambios de color y movimiento están amplificados.

La magnificación euleriana lineal amplifica los cambios de intensidad de los píxeles en referencia al tiempo, lo cual puede ser útil para detectar el flujo de sangre debajo de la piel en el rostro humano. Sin embargo, el algoritmo también puede producir nuevos videos con una relación señal ruido demasiado baja como para que la visualización sea precisa o confiable. Esto puede ser causado por dos factores, como consecuencia de señales con muy baja amplitud en relación al resto del contenido del video y la presencia de ruido que no puede ser removido. Esto se debe principalmente a que las

fotopletismografías remotas son susceptibles a artefactos provocados por movimiento que caen dentro del mismo rango de frecuencias que la frecuencia cardíaca (ver figura 3.15). Otra fuente de ruido posible son aquellas que provienen del propio dispositivo de captura del video.

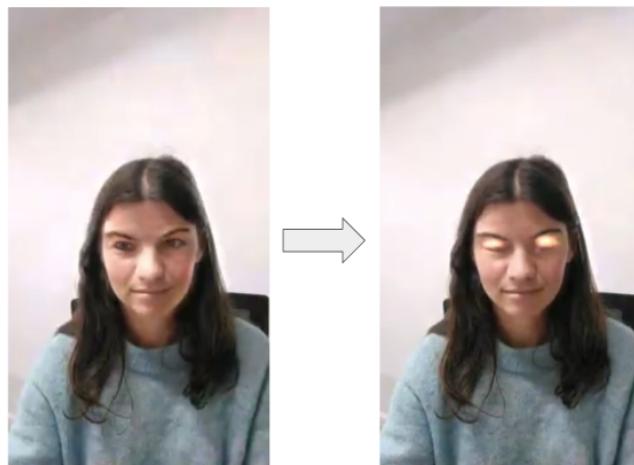


Figura 3.15: Magnificación del ruido generado por el parpadeo.

A pesar de estas limitaciones, se filtró con un pasabandas las señales cercanas a la frecuencia cardíaca y, luego, se amplificó la señal de los videos.

3.5.4. Entrenamiento de modelos para predicción de la frecuencia cardíaca

Tras el preprocesamiento y normalización de los datasets utilizados, detallado en las secciones previas, se entrenaron dos tipos de modelos de redes neuronales en 3 dimensiones. Cada video fue transformado en un arreglo de 4 dimensiones por la red neuronal: tiempo, las dimensiones de la imagen (ancho y alto), y la cantidad de capas. De estas dimensiones entre convolución a convolución con los filtros, solo el tiempo, el ancho y largo de la imagen se modificaron hasta la capa final.

Los dos modelos se realizaron construyéndose de cero debido a que no se encontraron modelos ya ajustados con sus pesos para realizar un fine-tuning. Los modelos se diferencian principalmente en el tipo de output: una predicción numérica del valor de frecuencia cardíaca para el primero, y una señal de FPG para el segundo.

Modelo tridimensional con valor numérico como salida

Como primer enfoque se buscó predecir la frecuencia cardíaca como valor numérico. La arquitectura se basó en el modelo propuesto de Physnet con algunas modificaciones en Yu et. al. [33] con Tensorflow, Keras. Physnet 3DCNN es un modelo espacio-

Nº Bloque	Capa	Formato de salida	Cantidad de parámetros
1	Conv3D	(None, 64, 64, 300, 16)	1216
1	Batch Normalization	(None, 64, 64, 300, 16)	64
1	Activación ReLU	(None, 64, 64, 300, 16)	0
2	MaxPooling3D	(None, 32, 32, 300, 16)	0
3	Conv3D	(None, 32, 32, 300, 32)	12832
3	Batch Normalization	(None, 32, 32, 300, 32)	128
3	Activación ReLU	(None, 32, 32, 300, 32)	0
4	Conv3D	(None, 32, 32, 300, 32)	27680
4	Batch Normalization	(None, 32, 32, 300, 32)	128
4	Activación ReLU	(None, 32, 32, 300, 32)	0
5	MaxPooling3D	(None, 16, 16, 150, 32)	0
6	Conv3D	(None, 16, 16, 150, 64)	55360
6	Batch Normalization	(None, 16, 16, 150, 64)	256
6	Activación ReLU	(None, 16, 16, 150, 64)	0
7	Conv3D	(None, 16, 16, 150, 64)	110656
7	Batch Normalization	(None, 16, 16, 150, 64)	256
7	Activación ReLU	(None, 16, 16, 150, 64)	0
8	MaxPooling3D	(None, 8, 8, 75, 64)	0
9	Conv3D	(None, 8, 8, 75, 64)	110656
9	Batch Normalization	(None, 8, 8, 75, 64)	256
9	Activación ReLU	(None, 8, 8, 75, 64)	0
10	Conv3D	(None, 8, 8, 75, 64)	110656
10	Batch Normalization	(None, 8, 8, 75, 64)	256
10	Activación ReLU	(None, 8, 8, 75, 64)	0
11	MaxPooling3D	(None, 4, 4, 75, 64)	0
12	Conv3D	(None, 4, 4, 75, 64)	110656
12	Batch Normalization	(None, 4, 4, 75, 64)	256
12	Activación ReLU	(None, 4, 4, 75, 64)	0
13	Conv3D	(None, 4, 4, 75, 64)	110656
13	Batch Normalization	(None, 4, 4, 75, 64)	256
13	Activación ReLU	(None, 4, 4, 75, 64)	0
14	Average Pooling3D	(None, 1, 1, 75, 64)	0
15	Flatten	(None, 4800)	0
16	Dense	(None, 1)	4801

Tabla 3.6: Estructura de la red convolucional 3D con output de frecuencia cardíaca como valor numérico.

temporal que adopta convoluciones con filtros de 3x3x3 para extraer las características semánticas de rFPG en el dominio espacial y temporal simultáneamente. Este modelo ayuda a aprender características de contexto más robustas, en este caso, para realizar en forma de regresión la predicción de la frecuencia cardíaca. Originalmente el modelo estaba inspirado en un codificador-decodificador, pero dado que el objetivo final era la predicción de un valor numérico escalar, se eliminaron las capas del decodificador y se modificó la capa final reemplazándola por una capa densa. La arquitectura completa se puede visualizar en la tabla 3.6. La cantidad de parámetros totales de esta red es 652365 de los cuales 651437 fueron entrenables.

El dataset UMA-rPPG fue utilizado con diferentes preprocesamientos. En una primera instancia, el dataset entrenó un modelo (Modelos 3DCNNv1.X) solo con detección de rostro aplicado. El modelo se entrenó con una partición del dataset para el conjunto de entrenamiento y validación de 0.8 y 0.2 respectivamente, con 252 videos de 10 segundos (300 frames) en un principio. Los modelos 3DCNNv1.X se diferencian por los distintos parámetros alterados en el entrenamiento, siendo X un número entero que indica cuál es la combinación seleccionada para su entrenamiento.

Dentro de los parámetros alterados para comparar las distintas versiones del modelo, en primer lugar se ajustó el ancho y largo del área de interés extraída con la detección de rostro para normalizar el tamaño del rostro detectado y ajustar su tamaño de acuerdo al consumo de memoria RAM. Se seleccionaron dos tipos de medidas. Para el modelo 3DCNNv1.0, se ajustaron 64 px × 64 px y 128 px × 128 px para los modelos restantes.

El entrenamiento se repitió modificando los pasos de preprocesamiento, siendo el dataset con magnificación euleriana y detección de rostro con Mediapipe (Modelo 3DCNNv2.X) el que produjo mejores resultados. De la misma manera que con los modelos 3DCNNv1.X se modificaron sus parámetros buscando minimizar las métricas del error y utilizando la misma proporción en la partición del dataset, aplicada a 196 video de 10 segundos (300 frames). Se repitió el ajuste de tamaño de frame correspondiendo 64 px × 64 px para el modelo 3DCNNv2.0 y 128 px × 128 px para los modelos restantes dentro de esta categoría.

La capacidad computacional de las GPUs disponibles en Colaboratory de Google, permitió entrenar la red con los siguientes hiperparámetros:

- El tamaño de batch fue de 16 para los modelos 3DCNNv1.0 y 3DCNNv2.0; y 8, para los restantes. El decremento del tamaño de batch se debe a la capacidad de procesamiento para poder cargar los videos con sus respectivos tamaños.
- El optimizador elegido fue Adam, variando su tasa de aprendizaje desde 0.001.
- La cantidad de epochs fueron entre 30 y 300.
- La función de pérdida, al tratarse de una regresión, fue el error cuadrático medio

(MSE). Adicionalmente, se agregaron como métricas de error del modelo el MAE y RMSE.

- Todos los modelos poseen la función de activación ReLU y llevan padding para mantener un tamaño consistente entre las capas. Para los modelos 3DCNNv1.3 y 3DCNNv2.3, las capas convolucionales 3D poseen Batch normalization (ver sección 2.5.7).

En la tabla 3.7, se listan la combinación de los parámetros utilizados en el entrenamiento de sus respectivos modelos.

Modelo	Tamaño de frame (px)	Tamaño de batch	Batch Normalization	Cantidad de epochs
3DCNNv1.0	64x64	16	No	30-100
3DCNNv1.1	128x128	8	No	30-100
3DCNNv1.2	128x128	8	No	30-200
3DCNNv1.3	128x128	8	Si	30-100
3DCNNv2.0	64x64	16	No	30-100
3DCNNv2.1	128x128	8	No	30-100
3DCNNv2.2	128x128	8	No	100-200
3DCNNv2.3	128x128	8	No	100-300
3DCNNv2.4	128x128	8	Si	30-100

Tabla 3.7: Parámetros de entrenamiento de los diversos modelos.

Además, para el entrenamiento de la red se utilizó un tipo de callback disponible en Keras[99]. En este caso particular, se utilizó el callback ModelCheckpoint. Este método permite guardar el modelo completo o sus pesos al finalizar un epoch dependiendo de si la métrica elegida presenta el mejor rendimiento posible para la red respecto a anteriores epochs. De esta forma al superar la mejor métrica elegida, los pesos del modelo se guardan. La métrica elegida para evaluar el mejor modelo en el checkpoint fue el RMSE en el conjunto de validación.

Los parámetros de las capas establecidos siguieron la estructura presentada en Yu et. al.[33] para el codificador. Los parámetros establecidos de las capas se pueden observar en la tabla 3.8. Además, todas las capas contuvieron padding temporal y espacial para mantener un tamaño consistente de capa a capa.

Se eligió la capa densa como capa agregada debido a que la predicción de la frecuencia cardíaca es un problema de regresión. En los problemas de regresión se busca predecir un valor continuo y la tarea de la capa final será mapear los valores de la capa anterior con el valor final numérico, la frecuencia cardíaca a través de una relación dada

Capa	N° Bloque	# Canales	Tamaño de Kernel/Pool	Stride
Conv3D	1	16	(1, 5, 5)	(1, 1, 1)
MaxPooling3D	2, 11	-	(2,2,1)	(2,2,1)
Conv3D	3	32	(1, 5, 5)	(1, 1, 1)
Conv3D	4	32	(3,3,3)	(1, 1, 1)
MaxPooling3D	5,8	-	(2,2,2)	(2,2,2)
Conv3D	6, 7, 9, 10, 12, 13	64	(3,3,3)	(1, 1, 1)
Average Pooling3D	14	-	(1, 1, 1)	(8,8,1)

Tabla 3.8: Parámetros de las capas de la red convolucional 3D con output de frecuencia cardíaca como valor numérico.

por la función de activación que en este caso es lineal. Es por esto que la capa densa es ideal para este tipo de tareas, puesto que calcula el valor numérico de la salida como

$$Output = f_{activation}(Kernel \cdot input + bias), \quad (3.2)$$

siendo $f_{activation}$ la función de activación, en este caso la función lineal. Los pesos del kernel y bias se ajustaron en el entrenamiento al igual que los de las otras capas y estos se usaron para realizar un producto escalar con los valores del input, los valores resultantes de la capa anterior.

Modelo tridimensional con señal de fotopletismografía como salida

Dado los resultados obtenidos y las decisiones tomadas en el entrenamiento de los primeros modelos, producto de la reducida cantidad de datos iniciales con el dataset generado, se buscó ampliar los datos de entrenamiento. Como se planteó en la sección 3.3.2, se realizó una búsqueda de nuevos datos para ampliar el dataset original, encontrando el dataset UBFC, los cuales poseían horas de videos con señales de fotopletismografías asociadas. El dataset UMA-rPPG no fue utilizado para entrenamiento a partir del hallazgo de estos nuevos datasets, sino como dataset de testeo dado que solo se encontraba etiquetado por un valor escalar y no la señal de fotopletismografía como etiqueta.

Las dimensiones finales de los 610 videos en ancho, largo de frame y largo del video fueron de $128 \text{ px} \times 128 \text{ px} \times 128 \text{ frames}$. Estas dimensiones también corresponden a las dimensiones necesarias en la capa de entrada de la red convolucional.

Tras la preparación de los datasets UBFC, se siguió la arquitectura basada en Physnet en Yu et. al. [33] antes mencionada en Tensorflow, Keras. En la tabla 3.9 se muestra la arquitectura lograda junto con la cantidad de parámetros para cada capa y el tamaño de la salida de cada capa. Tal como en el modelo anterior, la red convolucional busca extraer las características temporales y espaciales en el video simultáneamente (ver figura 3.16), inspirado en un codificador-decodificador temporal conocido como

Nº Bloque	Capa	Formato de salida	Cantidad de parámetros
1	Conv3D	(None, 128, 128, 128, 16)	1216
1	Batch Normalization	(None, 128, 128, 128, 16)	64
1	Activación ReLU	(None, 128, 128, 128, 16)	0
2	MaxPooling3D	(None, 128, 64, 64, 16)	0
3	Conv3D	(None, 128, 64, 64, 32)	12832
3	Batch Normalization	(None, 128, 64, 64, 32)	128
3	Activación ReLU	(None, 128, 64, 64, 32)	0
4	Conv3D	(None, 128, 64, 64, 32)	27680
4	Batch Normalization	(None, 128, 64, 64, 32)	128
4	Activación ReLU	(None, 128, 64, 64, 32)	0
5	MaxPooling3D	(None, 128, 32, 32, 32)	0
6	Conv3D	(None, 64, 32, 32, 64)	55360
6	Batch Normalization	(None, 64, 32, 32, 64)	256
6	Activación ReLU	(None, 64, 32, 32, 64)	0
7	Conv3D	(None, 64, 32, 32, 64)	110656
7	Batch Normalization	(None, 64, 32, 32, 64)	256
7	Activación ReLU	(None, 64, 32, 32, 64)	0
8	MaxPooling3D	(None, 32, 16, 16, 64)	0
9	Conv3D	(None, 32, 16, 16, 64)	110656
9	Batch Normalization	(None, 32, 16, 16, 64)	256
9	Activación ReLU	(None, 32, 16, 16, 64)	0
10	Conv3D	(None, 32, 16, 16, 64)	110656
10	Batch Normalization	(None, 32, 16, 16, 64)	256
10	Activación ReLU	(None, 32, 16, 16, 64)	0
11	MaxPooling3D	(None, 32, 8, 8, 64)	0
12	Conv3D	(None, 32, 8, 8, 64)	110656
12	Batch Normalization	(None, 32, 8, 8, 64)	256
12	Activación ReLU	(None, 32, 8, 8, 64)	0
13	Conv3D	(None, 32, 8, 8, 64)	110656
13	Batch Normalization	(None, 32, 8, 8, 64)	256
13	Activación ReLU	(None, 32, 8, 8, 64)	0
14	Conv3D transpuesta *	(None, 64, 8, 8, 64)	16448
14	Batch Normalization *	(None, 64, 8, 8, 64)	256
14	Activación eLU *	(None, 64, 8, 8, 64)	0
15	Conv3D transpuesta *	(None, 128, 8, 8, 64)	16448
15	Batch Normalization *	(None, 128, 8, 8, 64)	256
15	Activación eLU *	(None, 128, 8, 8, 64)	0
16	Average Pooling3D *	(None, 128, 1, 1, 64)	0
17	Conv3D	(None, 128, 1, 1, 1)	65
18	Flatten	(None, 128)	0

Tabla 3.9: Estructura de la red convolucional 3D con output de señal de FPG.

Physnet-3DCNN-ED, buscando reducir la redundancia y ruido temporal explotando el contexto temporal. Para dicha estructura se realizaron varias pruebas al inicializar los pesos aleatoriamente y la semilla de aleatoriedad para realizar las particiones de entrenamiento y validación. Dichas particiones fueron realizadas en un ratio 0,9-0,1, respectivamente. La cantidad de parametros totales de esta red fue de 685697, de los cuales solo 684513 fueron entrenables.

Si el objetivo de la red neuronal entrenada es predecir un valor único numérico como la frecuencia cardíaca a partir de un video, esto implicaría que la red debe reconocer el área de interés donde se encuentra la señal, segmentar dicha área, extraer la señal de fotopletismografía óptima y estimar la frecuencia a partir de ella. Esto es lo que se buscó con las anteriores redes presentadas. Sin embargo, con esta red, se limitó la cantidad de tareas que la red neuronal debía realizar a simplemente extraer la señal óptima.

Dada la capacidad computacional limitada que se poseía (16 GB de RAM y GPU provista por Colab), el tamaño de batch fue de 4, lo cual permitió entrenar la cantidad de parámetros que poseía la red y cargar los videos, una fuente de datos de gran tamaño, a través de un Batch Generator, derivado del Generator de Keras. El generador contiene un método de cargar los videos mediante el path de cada archivo, mientras que las señales de rFPG se asocian a ellos mediante un dataframe de la librería Pandas.

La función elegida como función de pérdida para ajustar el modelo en el entrenamiento y evaluar el mejor modelo dentro del conjunto de validación fue la correlación de Pearson negativa [33] (ver figura 3.16). Esta función permite maximizar la similitud entre la señal predicción y la señal original provista por los datasets UBFC. Siendo el valor máximo 1 (ver sección 2.6.3), cuanto mayor sea la métrica, más similares serán las señales. La similitud entre las señales busca maximizar la similitud en la tendencia y disminuir el error del posicionamiento de picos y, por lo tanto, disminuir el error en el cálculo de parámetros fisiológicos (en este caso, de la frecuencia cardíaca).

En el entrenamiento de la red, diversas particiones del dataset e inicialización de los pesos de la red fueron probados hasta encontrar la combinación que diese los mejores resultados. Al igual que para el modelo anterior, también se utilizó el callback de ModelCheckpoint, pero esta vez en el valor de la función de pérdida en el conjunto de validación como métrica de seguimiento.

Cada modelo fue entrenado desde 100 a 400 epochs hasta obtener la mejor métrica posible. Las funciones de activación para las capas convolucionales fueron ReLU, salvo por las dos últimas que fueron definidas como eLU (ver la tabla 3.9). El optimizador elegido fue Adam, siendo su tasa de aprendizaje variada entre 0,01 y $1e - 4$.

La estructura de la red sigue también los lineamientos planteados en Yu et. al. [33]. La mayoría de los parámetros se mantuvieron del modelo anterior y para su mejor visualización se encuentran resumidos en la tabla 3.10. En la parte del codificador pre-

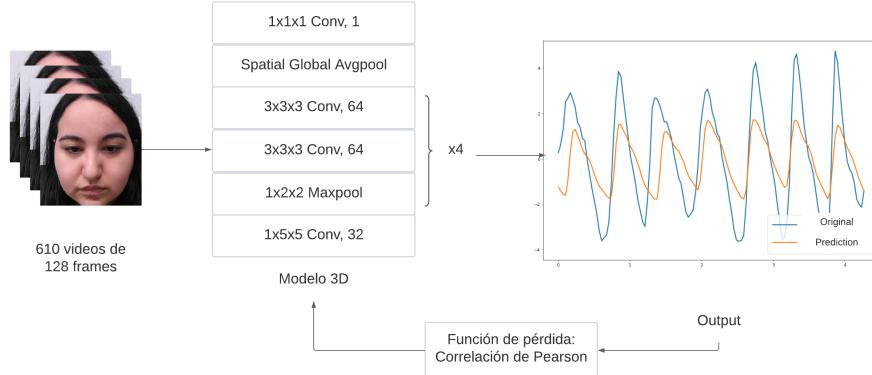


Figura 3.16: Diagrama de flujo de entrenamiento de la red tridimensional con output de fotopletismografía.

Capa	Nº Bloque	# Canales	Tamaño de Kernel/Pool	Stride
Conv3D	1	16	(1, 5, 5)	(1, 1, 1)
MaxPooling3D	2, 11	-	(1,2,2)	(1,2,2)
Conv3D	3	32	(1, 5, 5)	(1, 1, 1)
Conv3D	4	32	(3,3,3)	(1, 1, 1)
MaxPooling3D	5,8	-	(2,2,2)	(2,2,2)
Conv3D	6,7,9,10, 12, 13	64	(3,3,3)	(1, 1, 1)
Conv3D Transposed	14, 15	64	(4, 1, 1)	(2, 1, 1)
Average Pooling3D	16	-	(1, 1, 1)	(1,8,8)
Conv3D	17	1	(1, 1, 1)	(1, 1, 1)

Tabla 3.10: Parámetros de las capas de la red convolucional 3D con output de señal de fotopletismografía.

senta una estructura similar a la del primer modelo hasta el bloque 13, pero también se le agrega un decodificador al final. La principal diferencia en el codificados se encuentra en el poolsize y stride de las capas de Max Pooling y AveragePooling. Estos parámetros ayudan a la reducción de las dimensiones. Es consistente la diferencia dado que en el anterior modelo se buscaba encontrar un único valor numérico mientras que en este modelo se busca mantener la temporalidad al final del modelo. Para ello, se utiliza un decodificador conformado por capas deconvolucionales (marcadas con * en la tabla 3.9) con las cuales se buscó volver a la misma dimensión temporal inicial, 128 frames.

Una vez entrenado y elegido el mejor modelo (3DCNNv3), se testeó el modelo con el dataset UMA-rPPG. Sobre este conjunto de datos se procedió a realizar las predicciones de las señales y a realizar un postprocesamiento para calcular la frecuencia cardíaca de la señal original y la señal predicha. Con ambos valores de frecuencia cardíaca, se calcularon los errores de MAE y RMSE para poder compararlos con el primer modelo logrado.

3.5.5. Postprocesamiento de la señal de fotopletismografía como output

El modelo 3DCNNv3 obtiene la señal de fotopletismografía como salida. Estas señales y las señales que sirven como etiquetas del modelo de los datasets de UBFC deben ser procesadas para poder inferir la frecuencia cardíaca (ver figura 3.17).

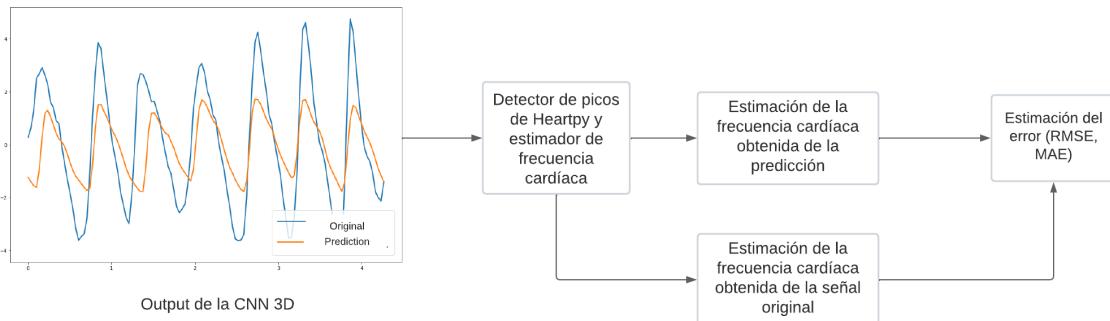


Figura 3.17: Postprocesamiento y comparación de los modelos

Al obtener la predicción, el algoritmo detector de picos de la librería Heartpy estima la frecuencia cardíaca. Este mismo ha demostrado mejor rendimiento que otros algoritmos de estado del arte en datos de FPG[100]. Se obtuvo una media de error de posicionamiento de picos más baja (de 0,89 milisegundos), una estimación de la frecuencia más robusta debido al bajo ratio de picos posicionados de forma correcta y un RMSE de la frecuencia de un valor de 3,77 lpm. Son por estas razones que el algoritmo fue elegido por sobre otros para completar la inferencia de la frecuencia cardíaca partir de las señales obtenidas del modelo. Adicionalmente, el algoritmo verifica la calidad de la señal, devolviendo un error si esta no contiene una señal adecuada. Este paso permitió detectar un grado de confiabilidad en el resultado del modelo, puesto que la predicción de la FPG es lo que nos permite calcular la frecuencia cardíaca.

3.5.6. Métodos de evaluación

Para la evaluación de los modelos en tres dimensiones con un valor escalar como output (3DCNNv1.X y 3DCNNv2.X), se utilizó el RMSE para determinar el mejor modelo dentro de los entrenados y, como métrica secundaria, el MAE. Tanto el RMSE como el MAE son métricas típicas a la hora de evaluar modelos de regresión que indican la distancia entre la predicción y el valor real (en este caso, de la frecuencia cardíaca). La diferencia sustancial entre ambas métricas está en que el RMSE es menos robusta al ser afectada por valores atípicos (*outliers*) del conjunto debido a que se eleva al cuadrado el cálculo de error.

Además, con el fin de maximizar la cantidad de datos provista para el entrenamiento y validación, no se realizó una partición de testeo. Las opciones para evaluar estos modelos resultaron en dos alternativas: volver a adquirir muestras o buscar datos disponibles en internet. Se optó por la segunda alternativa, la cual permitió testear los modelos v1 y v2 con el dataset de UBFC-rPPG. Para poder calcular las métricas de error, se calcularon las frecuencias cardíacas que etiquetaron los videos del dataset a partir de las señales de FPG originales que se encontraban asociadas a estos. Se buscó analizar el comportamiento del modelo frente a frecuencias cardíacas normales, dado que los modelos fueron entrenados con datos mayoritariamente de frecuencia normal. Con el fin de limitar dicha distribución de forma que fuese similar a la de los datos de entrenamiento, se muestreó los datasets de testeo en los rangos correspondientes. Estos submuestreos realizados fueron denominados como datasets de test acotado. Esta limitación luego se repitió para los modelos restantes de acuerdo a sus datos de entrenamiento.

Para el modelo tridimensional con la fotopletismografía como salida (3DCNNv3), como se ha establecido previamente, la métrica de elección del mejor modelo fue la correlación de Pearson dado que se busca que las señales predichas sean lo más similares a las señales originales. Los resultados provistos en Yu et. al. [32] demuestran que dicha función de pérdida posee un rendimiento superior en comparación con el MSE. Esto se debe a que MSE considera la amplitud de las señales dentro del error, una característica de la señal irrelevante para el cálculo de la frecuencia cardíaca, que solo considera la cantidad de picos de la señal.

La señal de FPG no permitía comparar con los modelos cuyo output era un único valor numérico. Sin embargo, con las frecuencias obtenidas mediante el postprocesado, fue posible calcular el RMSE y MAE y comparar con el primer modelo tridimensional con un valor escalar numérico como salida. Estas tres métricas se utilizaron para poder comparar los dos tipos de modelos explicados en las secciones previas. Para evaluar el desempeño del modelo en condiciones similares a las de entrenamiento se calcularon estas métricas para la sección del dataset apartado para validación. En una instancia posterior, se puso a prueba la robustez del algoritmo evaluándolo con los videos del dataset UMA-rPPG. A los videos de 10 segundos se los preprocesó realizando un ventaneo de 128 frames con superposición de 15 frames para obtener un promedio de las predicciones finales de frecuencia cardíaca. De los 273 videos, en 233 se detectó la cara y estos fueron utilizados para el testeo. El dataset de UMA-rPPG únicamente contiene el valor numérico de frecuencia cardíaca por lo cual no pudo compararse las FPG predichas con señales originales pero sí el valor estimado de frecuencia cardíaca a partir de la frecuencia predicha con el valor del oxímetro. Este dataset permitió evaluar como generalizaba el modelo en condiciones diferentes a las de entrenamiento.

Adicionalmente, se realizó una prueba estadística de Pearson sobre los resultados

en validación y test de los 3 modelos para verificar si existe una relación monótona entre las predicciones de frecuencia cardíaca del modelo y de los valores medidos por el oxímetro comercial.

Capítulo 4

Resultados y discusión

4.1. Métodos aplicados a los videos obtenidos del dedo índice de los sujetos.

4.1.1. Primeros modelos de predicción de frecuencia cardíaca

Resultados del entrenamiento

Modelo	RMSE entrenamiento (lpm)	MAE entrenamiento (lpm)	RMSE validación (lpm)	MAE validación (lpm)
1DCNNv1.1	99.87	96.081	90.374	87.481
1DCNNv1.2	27,477	20.823	28.7404	21.8883
1DCNNv1.3	11.062	7.832	12.398	8.968
1DCNNv1.4	10.46	7.292	10.7404	7.875
1DCNNv1.5	12.43	7.822	10.605	8.313
1DCNNv1.6	9.436	5.785	10.846	7.139

Tabla 4.1: Metricas de entrenamiento y validación de primeros modelos convolucionales 1D entrenados

Como se muestra en la tabla 4.1, los modelos con mejores resultados se lograron con un learning rate entre 0,01 y 0,001, resultando el mejor modelo el entrenado con un learning rate de 0,008 (modelo 1DCNNv1.4).

Resultados del Testeo

Se evaluaron los 3 mejores modelos (1DCNNv1.3, 1DCNNv1.4 y 1DCNNv1.6) con el dataset de testeo.

Modelo	RMSE testeo (lpm)	MAE testeo (lpm)
1DCNNv1.3	627.883	263.127
1DCNNv1.4	710.543	289.232
1DCNNv1.6	681.212	269.345

Tabla 4.2: Metricas de entrenamiento y validación de primeros modelos convolucionales 1D entrenados

A partir de los resultados mostrados en la tabla 4.2, se pudo ver que ninguno de los modelos logró generalizar los resultados para el dataset de testeo.

El error (MAE) al presentar datos de un dataset de dominio distinto al de entrenamiento resultó en todos los casos mayor a 200 lpm, lo que es en sí mayor a una frecuencia cardíaca normal. Por lo tanto, ninguno de estos modelos lograría realizar buenas predicciones en condiciones de captura diferentes a las del dataset utilizado para entrenar.

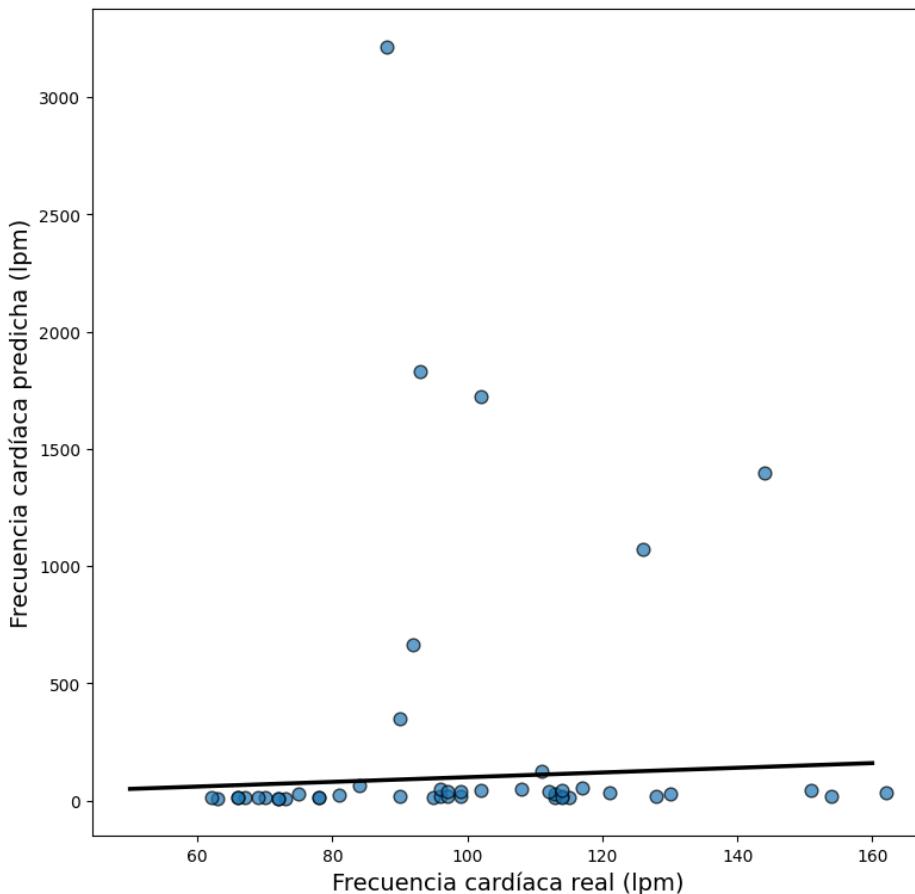


Figura 4.1: Frecuencias cardíacas predichas por modelo 1DCNN-v1.3 en función de las frecuencias cardíacas reales

En la figura 4.1 se muestran las predicciones realizadas por uno de los tres modelos

(1DCNN-v1.3). Se observó que en la gran mayoría de los casos el modelo predijo valores mucho mayores (cercaos a los 1000 lpm) o bien mucho menores (cercaos a los 30 lpm) a los valores reales. Estos valores fueron lejanos a los valores normales de frecuencias cardíacas humanas fisiológicas, por lo que se concluyó que estos modelos no podrían ser utilizados para la predicción de frecuencia cardíaca.

A partir del análisis realizado en la sección 3.4.4 se llegó a la conclusión de que la incapacidad de generalización se debió a las condiciones tan controladas de captura del dataset utilizado para el entrenamiento de estos modelos.

Todos los videos utilizados para el entrenamiento de los modelos fueron capturados con el mismo dispositivo. Es por esto que para todos ellos la señal de FPG se encuentra en el mismo canal, tal como se muestra en la figura 3.6. Como consecuencia, estos modelos 'aprendieron' únicamente a encontrar patrones en dicho canal, y al intentar realizar una predicción a partir de un video capturado con otro dispositivo, cuya señal de FPG se encuentra en otro canal, estos no lograron hacer buenas predicciones.

4.1.2. Modelo clasificador de señales

Para los métodos asociados a los videos del dedo índice, se analizó el desempeño del clasificador de señales de FPG, cuyos resultados se muestran en la tabla 4.3.

Modelo	Exactitud	Precisión	Exhaustividad	F1 Score
ResNet50-based	0.90	0.90	0.86	0.88
VGG16-based	0.88	0.86	0.86	0.86

Tabla 4.3: Métricas de validación de los modelos clasificadores de señales

Ambos modelos presentan una exhaustividad igual, pero el modelo basado en ResNet50 presenta mejor exactitud, precisión y *F1-score*, por lo que se tomó este como el mejor modelo.

También se calculó la proporción de descarte de videos por el modelo clasificador a partir del dataset de testeо. De los 40 videos el modelo clasificó 4 videos como si no contuviesen ninguna señal de FPG, es decir, 10 % de los videos contaban con una mala captura y deberían ser tomados nuevamente.

De los 4 videos descartados, 2 estaban asociados a frecuencias normales (menores a 100 lpm) y 2, a frecuencias taquicárdicas. De modo que la proporción de videos descartados por el modelo clasificador en ambos casos fue similar y se concluyó que la calidad de captura de las señales y la capacidad del modelo clasificador de identificar señales de FPG no presentaban sesgos considerables en relación a una situación taquicárdica.

4.1.3. Modelo de predicción de la frecuencia cardíaca

Resultados del entrenamiento

Modelo	RMSE entrenamiento (lpm)	MAE entrenamiento (lpm)	RMSE validación (lpm)	MAE validación (lpm)
1DCNNv2.1	6.37	4.78	13.22	9.28
1DCNNv2.2	10.71	7.51	11.54	8.33
1DCNNv2.3	9.79	6.76	11.14	7.70
1DCNNv2.4	9.80	6.89	11.08	7.58
1DCNNv2.5	3.92	2.94	12.64	8.49
1DCNNv2.6	7.29	5.19	3.23	11.69

Tabla 4.4: Métricas de los modelos con *output* de frecuencia cardíaca para los datos de entrenamiento y validación con la toma del dedo índice: MAE y RMSE entre la frecuencia original y la predicción.

En la tabla 4.4 se muestran los resultados para cada modelo entrenado para obtener frecuencia cardíaca con videos del dedo. Para los datos de entrenamiento de UMA-dPPG, se observó que los modelos que poseían menor error con los datos de entrenamiento fueron 1DCNNv2.5 como el mejor modelo, 1DCNNv2.1 como el segundo mejor y v6 como el tercero. No obstante, frente a los datos de validación, las métricas de error aumentaron 94,14 %, 188,77 % y 125,2 % para el MAE, respectivamente. Con la métrica de RMSE, el único modelo que tuvo una disminución del error de un 55,69 % fue el 1DCNNv2.6; esta mejora no se reflejó en el MAE, por lo cual, se lo descartó como un posible mejor modelo. Por otro lado, los modelos 1DCNNv2.1 y 1DCNNv2.5 presentaron un aumento del RMSE del 107,54 % y del 222,45 %, respectivamente. Los resultados obtenidos para estos modelos indicaron una presencia de sobreajuste del modelo a los datos de entrenamiento. Es por esta razón que los modelos 1DCNNv2.1, 1DCNNv2.5 y 1DCNNv2.6, no fueron seleccionados como mejores modelos.

Entre los modelos restantes entrenados, el mejor resultado obtenido fue con un MAE de 7.58 lpm y un RMSE de 11.08 lpm en la partición de validación, y corresponde al modelo con filtros de *kernel* tamaño 7, el modelo v4 (ver tabla 4.4). En comparación a sus métricas de error con los datos de entrenamiento, hubo un aumento de error del 14,1 % para el RMSE y 10,01 % para el MAE con los datos de validación, significativamente menor a los modelos con mejores métricas.

Además, en términos generales, se observó que los modelos entrenados con señales a las que se le aplicaron filtros de extracción de ruido (modelos 1DCNNv2.2, 1DCNNv2.3, 1DCNNv2.4 y 1DCNNv2.6) arrojan mejores resultados de validación y sobreajustan menos que aquellos que fueron entrenados con señales sin filtrar (modelos 1DCNNv2.1 y 1DCNNv2.5).

Resultados de test

A partir de los datos obtenidos de los diferentes modelos, se seleccionó el mejor entre ellos (1DCNNv2.4) para ser evaluado en el nuevo dataset de testeo recolectado. Evaluando el modelo, el MAE resultó de 7.02 lpm, y el RMSE de 10.01 lpm. Estos valores fueron similares a los resultados obtenidos en la partición de validación indicando una buena generalización del modelo frente a nuevos datos obtenidos bajo diferentes condiciones de captura (iluminación, dispositivo, entre otros).

Por otro lado, se buscó analizar la correlación entre las frecuencias predichas y las frecuencias medidas por el oxímetro calculando el coeficiente de correlación cuadrado (R^2). Este resultó de 0,84, por lo que se concluyó que el modelo tiene un buen poder predictivo.

En la figura 4.2 se muestran las frecuencias cardíacas predichas en función a las reales para el dataset de testeo. Se observó que para frecuencias más bajas el modelo predijo valores por arriba del valor real, mientras que para frecuencias más altas el modelo predijo una frecuencia por debajo del valor real.

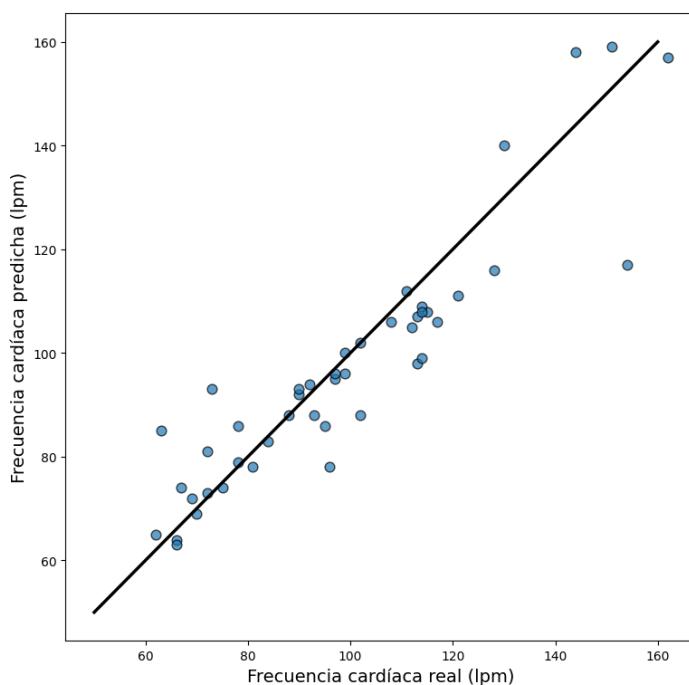


Figura 4.2: Frecuencias cardíacas predichas por modelo 1DCNNv2.4 en función de las frecuencias cardíacas reales

Un hecho a resaltar es el ligero aumento del error en los casos en los que la frecuencia cardíaca superaba los 100 lpm. Es posible que esto se deba a que el modelo fue entrenado con mayor cantidad de señales de frecuencias fisiológicas normales que taquicárdicas, mientras que para el testeo se utilizaron datos distribuidos uniformemente, como se muestra en la figura 4.3.

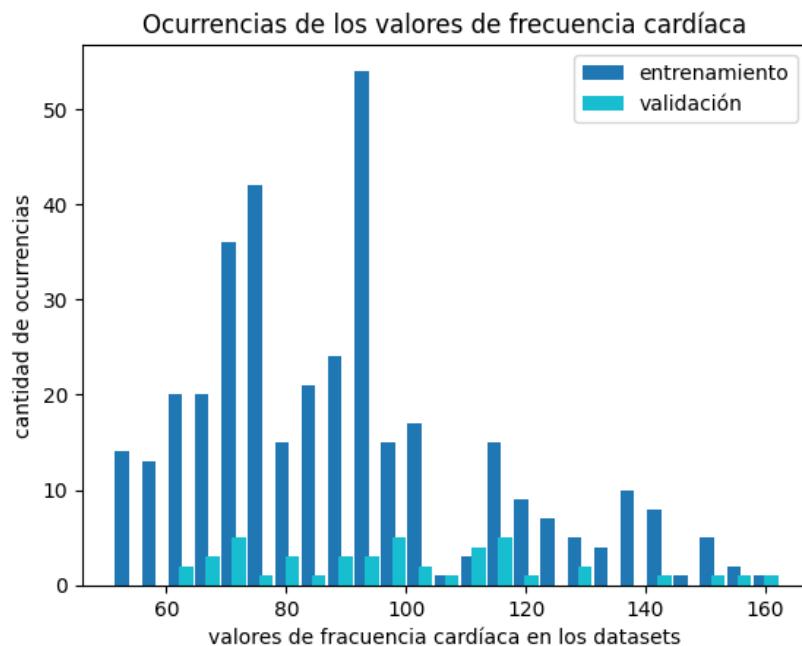


Figura 4.3: Distribución de frecuencias cardíacas medidas por el oxímetro en el dataset de entrenamiento (en azul) y testeо (en cian) de los modelos entrenados con videos del dedo

4.2. Métodos aplicados a los videos obtenidos de la cara de los sujetos

4.2.1. Modelos de predicción de frecuencia cardíaca como valor numéricico escalar

Resultados del entrenamiento

Los primeros resultados analizados fueron aquellos del modelo 3D-CNN en sus dos versiones (3DCNNv1.X y 3DCNNv2.X) y variaciones de hiperparámetros (ver tabla 3.7), los cuales poseen una salida escalar que corresponde al valor numérico de la frecuencia cardíaca.

Dentro de las métricas de entrenamiento (ver tabla A.5), los modelos 3DCNN v2.X mostraron un MAE menor a 5 lpm y un RMSE menor a 5,2 lpm salvo por el modelo 3DCNNv2.2. Dentro de las diferentes versiones, los modelos 3DCNNv2.3, v2.0, v1.0 y v2.4 fueron aquellos con menor error (ordenados de menor a mayor error).

No obstante, al calcular las métricas con el conjunto de validación (ver tabla A.6), el error de los modelos superó 8 lpm tanto para MAE como RMSE. Solo el modelo 3DCNNv2.2 no presentó un aumento de sus métricas de error, sino que estas disminuyeron con una diferencia de 0,2549 en el MAE y 0.7617 para el RMSE. Adicionalmente, las frecuencias cardíacas obtenidas por el oxímetro de pulso y aquellas obtenidas como predicción del modelo obtuvieron correlaciones fuertes, monótonas crecientes y

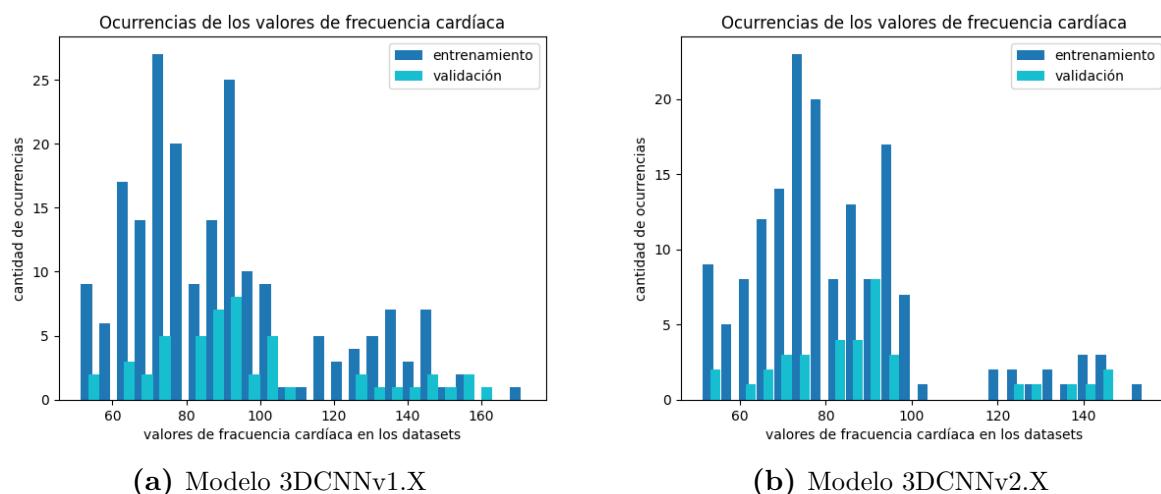


Figura 4.4: Distribución de frecuencias cardíacas medidas por el oxímetro en el dataset de entrenamiento (en azul) y validación (en cian) de los modelos con *output* de valor escalar.

lineales con valores mayores a 0.8 en todos los modelos con significancia estadística ($p\text{-value} < 0.05$). Los modelos 3DCNN v2.0, v2.2 y v1.1 son aquellos con menor MAE en orden creciente de dicho error; mientras que observando el RMSE, los modelos con menor error son v2.0, v2.2 y v2.1 los cuales también poseen los valores de correlación más elevados (> 0.9). La relación entre las frecuencias cardíacas obtenidas por medición y las predicciones puede observarse en la figura A.1 del Anexo para cada modelo con el conjunto de validación. Con los resultados obtenidos en esta etapa, los 4 modelos seleccionados (v2.0, v2.1, v2.2 y v1.1) por su menor error tienen la misma característica de tener un tamaño de frame de 128x128 px.

Un factor que afectó los resultados entre los modelos 3DCNNv1.X y 3DCNNv2.X fue la muestra de datos tomada (ver figura 4.4). La misma poseía una cantidad de datos limitada para los modelos de predicción de la frecuencia cardíaca con videos de la cara, por lo cual no abarcó un amplio espectro de casos. Dentro de las muestras, un tercio de los datos se correspondían con taquicardia, los valores que idealmente deberían identificarse para la detección de patologías. La primera diferencia entre los dos tipos modelos utilizados y sus resultados yace en el dataset utilizado para entrenar cada uno. Como se puede observar en la figura 4.4, el modelo 3DCNNv1.X fue entrenado con 261 datos los cuales poseían un rango de valores más amplio en frecuencias cardíacas (50 a 170 lpm), mientras el modelo 3DCNNv2.X solo con 196 muestras, con una variación de 50 a 150 lpm. Esto se debe a que aquellos videos que no pudieron ser procesados con la magnificación euleriana fueron descartados, sin embargo, se mantuvo la partición de entrenamiento-validación lo más parecida posible entre ambos modelos.

Resultados del test

Se analizaron todos los modelos con el dataset UBFC como test (ver tabla A.7) y test acotado para realizar una comparación entre cada uno de ellos con datos que no hayan sido utilizados en su etapa de entrenamiento.

Dentro de los modelos 3DCNNv1.X, el modelo con más error fue el modelo v1.0 incluso al acotar el dataset a muestras menores de 100 lpm. La decisión de restringir los valores dentro del dataset de muestreo se basó en la diferencia entre los errores de cada categoría (test y test acotado) la cual está relacionada a la cantidad de muestras de cada uno en el dataset de UMA-rPPG utilizado para el entrenamiento. En el caso de UMA-rPPG, las muestras se encuentran concentradas en los valores por debajo de 100 lpm. En cambio, el dataset de UBFC utilizado para el testeo presenta una amplia distribución de frecuencias cardíacas tanto para la partición de entrenamiento como para la de validación. Se observó que el dataset elegido para validar el modelo con *output* de señal contiene una gran cantidad de muestras taquicárdicas (ver figura 4.5a).

La principal diferencia entre el modelo v1.0 y el resto de 3DCNNv1.X yace en el tamaño del frame utilizado. La pérdida de información ocurrida al realizar el cambio de dimensión del rostro detectado podría afectar directamente al desempeño del modelo. El segundo peor modelo, v1.2, presenta un error mayor a los 50 lpm y corresponde al modelo que fue entrenado por más de 100 epochs por lo cual aumenta la probabilidad de sobreajuste sobre los datos de entrenamiento. El mejor modelo de 3DCNNv1.X posee un MAE de 33.2291 y un RMSE de 36.8576 lpm con el dataset completo; y un MAE de 18.3018 y un RMSE de 20.5675 lpm con el dataset acotado. Estos modelos fueron descartados no solo por su elevado error sino dado que la correlación obtenida entre las predicciones y las frecuencias tomadas como etiqueta no demostró una relación estadística (ver figura A.2 para el conjunto de datos de test completo y A.3 para el conjunto acotado).

Al aplicar la magnificación euleriana, el grupo de modelos de 3DCNNv2.X mostró una leve mejora en sus métricas de error con respecto al grupo anterior. El primer modelo en ser descartado fue el 3DCNNv2.3, único en poseer un error mayor a 35 lpm con el dataset de testeo completo y entrenado durante 300 epochs. Esto indicó una fuerte presencia de sobreajuste por lo cual se decidió sólo entrenar los siguientes modelos entre 100 y 200 epochs. El siguiente modelo en ser descartado fue el modelo 3DCNN v2.1. Las métricas de error en relación al v2.0 eran mucho más elevadas incluso en el dataset de testeo acotado (RMSE=15.42 lpm; MAE=19.87 lpm) aunque la correlación entre frecuencias era mayor e inversa ($\rho = -0,26$).

Entre los modelos restantes, el modelo v2.4 obtuvo un RMSE de 13.8541 lpm. A dicho modelo se le adicionó capas de Batch Normalization como método de preven-

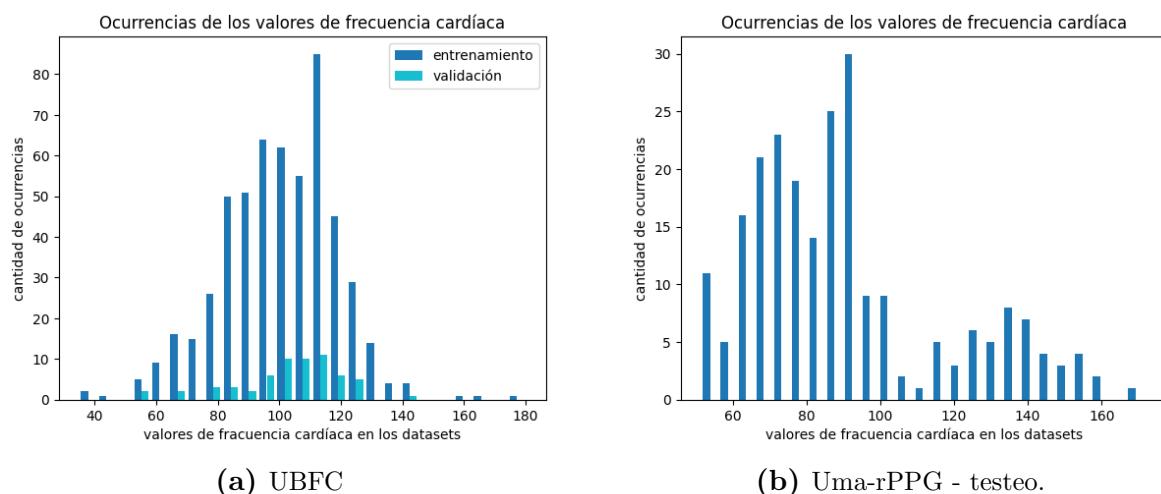


Figura 4.5: Distribución de frecuencias cardíacas (a) calculadas a partir de los FPGs originales en el dataset de entrenamiento y validación de los modelos con *output* de señal; (b) medidas con un óxímetro de pulso para el dataset de Uma-rPPG.

ción del sobreajuste en su entrenamiento lo cual podría explicar que se obtuviese el menor RMSE dentro de los resultados de testeo. A pesar de dicha mejora, se obtuvo un $p\text{-value} > 0,05$, rechazando la hipótesis nula de una relación monótona entre las frecuencias, por lo que fue descartado.

Tanto el modelo v2.2 como el v2.0 poseían ventajas de entre todos sus predecesores. Por un lado, el modelo v2.3 mantenía un error mayor que el v2.0 en ambas métricas, pero su correlación era más fuerte con un coeficiente de -0.3447 aunque inversa con significancia estadística. Al analizar la relación entre las frecuencias (ver figura A.3h) se observó que las frecuencias de oxímetro mayores a 90 se predijeron en sus videos como valores en el rango entre 60 y 90, donde se encuentran acumuladas la mayoría de las muestras de entrenamiento y validación de los modelos. Dado lo analizado, el modelo v2.0 fue seleccionado al poseer un menor error tanto para el dataset completo como el dataset acotado y mantener significancia estadística en el test de Pearson aunque con una relación monótona débil ($\rho = 0,19$).

4.2.2. Modelos de predicción de la señal de fotopletismografía

Resultados del entrenamiento

En una segunda instancia, el valor máximo alcanzado para el modelo de predicción de FPG fue de $\rho = 0,9206$ con el dataset de entrenamiento y $\rho = 0,8244$ con el dataset de validación para la métrica de correlación de Pearson entre las señales originales y las predicciones.

En la figura 4.6 se puede observar los tres mejores y los tres peores resultados de la métrica de correlación de Pearson en el conjunto de validación. Los peores resultados de la correlación de Pearson se encontraron entre 0.4 y 0.5; mientras que los mejores

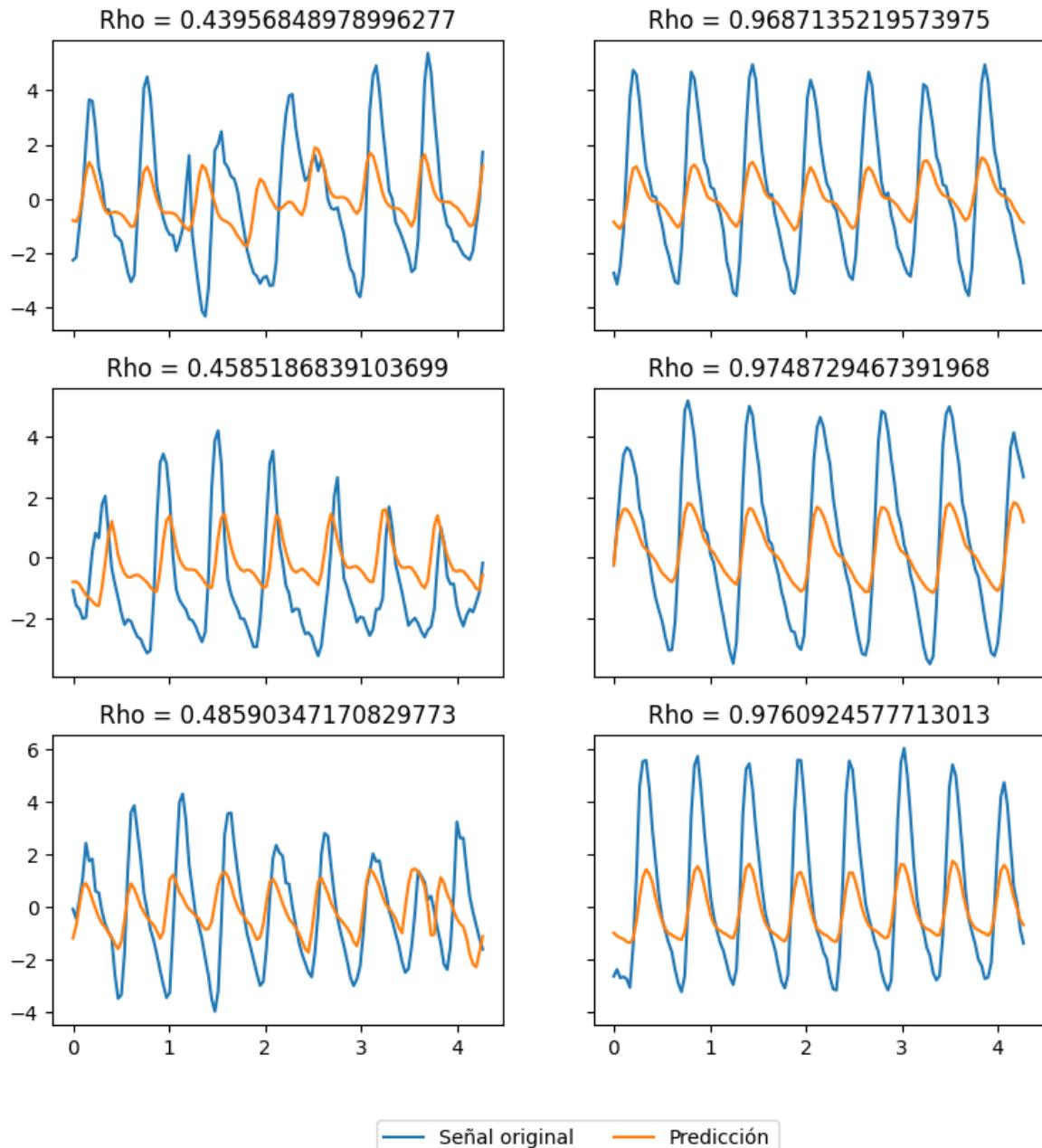


Figura 4.6: Mejores 3 y peores 3 resultados de la predicción de la señal de fotopletismografía en comparación a la señal original.

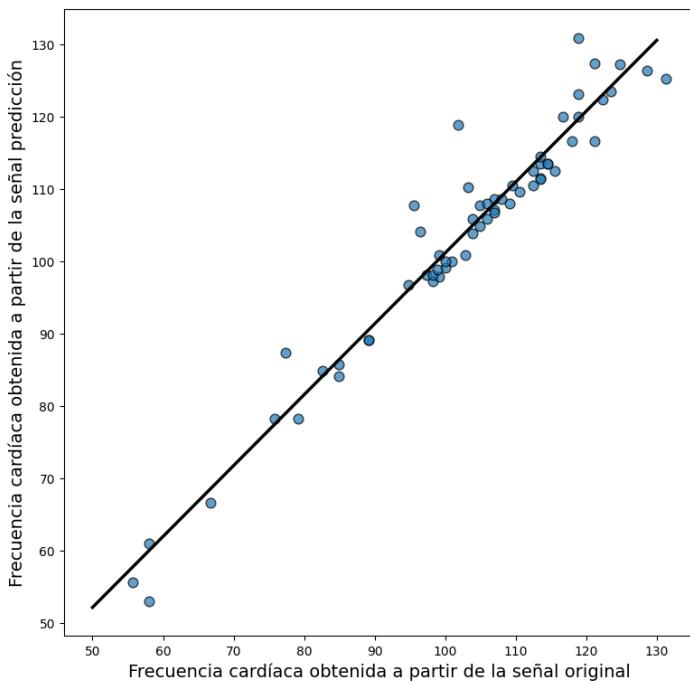


Figura 4.7: Frecuencia cardíaca estimada a partir de la señal predicción en función de las frecuencias estimadas a partir de la señal original.

superaron 0,95. Si bien el mal calculo en el posicionamiento de los picos no afecta dramáticamente el calculo de la frecuencia cardíaca, si puede afectar otros parámetros fisiológicos que se quieran agregar dentro del algoritmo a futuro.

Con las estimaciones de las frecuencias cardíacas a partir de ambas señales, se analizaron los resultados finales del flujo, a través del RMSE y MAE. Para el conjunto de validación, los resultados obtenidos fueron favorables, con un RMSE de 4,12 lpm y un MAE de 2,43 lpm.

A las frecuencias calculadas también se les aplicó el test estadístico de Pearson. En el caso del conjunto de validación, el p-value del test de Pearson fue $< 2,2e - 16$, es decir, menor al nivel de significancia ($< 0,05$). Por lo tanto, la correlación entre las frecuencias de la predicción y de los datos originales es significativa y es fuerte dado que el coeficiente de Person es 0,971, cercano a 1. Esto indica una relación quasi-lineal y monótona entre los datos de la señal predicción y la señal original (ver figura 4.7).

Resultados del test

La distribución de los valores de frecuencia cardíaca del dataset de testeo utilizado pueden ser observados en la figura 4.5b, teniendo una mayoría de frecuencias normales (173 videos) en relación a las taquicárdicas (60 videos). Frente a estos datos, el modelo no presentó el rendimiento esperado, dado que sus métricas de MAE y RMSE fueron 24,6309 y 29,7269 lpm, respectivamente. Los valores calculados por el modelo en relación a los valores originales obtenidos con un oxímetro de pulso se muestran en la figura

Modelo	Dataset	MAE (lpm)	RMSE (lpm)	ρ	p-value
3DCNNv3	Validación	2,432	4,120	0,971	$< 2,2e - 16$
3DCNNv3	Test	24,631	29,727	0,1517	0,0205
3DCNNv3	Test - acotado	13,077	15,889	0,2953	0,0012

Tabla 4.5: Métricas de los modelos a comparar a través de la frecuencia cardíaca de la predicción contra los datos originales.

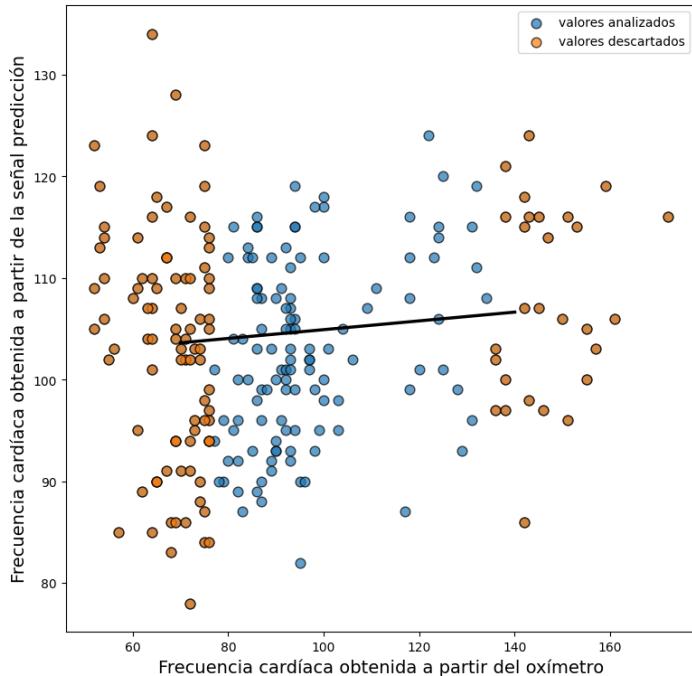


Figura 4.8: Frecuencia cardíaca estimada a partir de la señal predicción en función de las frecuencias obtenidas a partir del oxímetro comercial. Los valores con marcadores naranjas corresponden a los valores que se descartaron para un análisis más profundo de los resultados.

4.8. Este aumento del error se lo atribuyó a un sobreajuste del modelo sobre los datos con los que fue entrenado, aunque se mantiene una relación monótona de forma débil con un coeficiente de correlación de Pearson igual a 0,1517 con significancia estadística dado que el p-value del test de correlación fue menor a 0.05 (ver tabla 4.5).

Para poder tener una comprensión de los resultados se procedió a analizar el dataset utilizado en el entrenamiento y validación en relación al dataset de testeо. El primer parámetro analizado fue el valor de frecuencia cardíaca original, ya sea calculado a partir de la señal de FPG original para el dataset UBFC (ver figura 4.5a) u obtenido por el oxímetro para el dataset UMA-rPPG (ver figura 4.5b). Observando ambas figuras (4.5a y 4.5b) la diferencia notable se encuentra en el rango de valores de frecuencias cardíacas con mayor cantidad valores. Para el dataset de entrenamiento y validación, la mayor cantidad de muestras se encontraban en el rango entre 77 y 135 lpm. Al limitar el dataset de testeо a este rango de valores (Test acotado), se observó una disminución en ambos errores (13,0769 lpm para el MAE y 15,8896 para el RMSE), mientras que la correlación de Pearson aumentó a 0,2953 con un p-value de 0,0012, estadísticamente

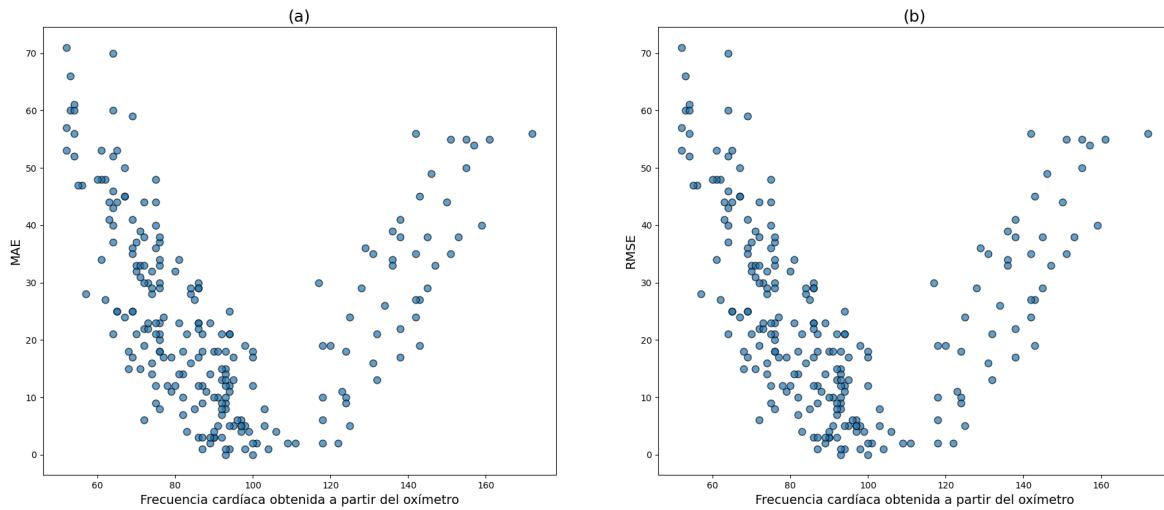


Figura 4.9: MAE (a) y RMSE (b) en función de las frecuencias obtenidas a partir del oxímetro comercial.

significativo (ver figura 4.9).

Si bien para el último caso se observó una disminución en el error, se observó aún la presencia de sobreajuste del modelo a los datos de entrenamiento. Esto puede deberse a las condiciones de iluminación en la toma del video y las características del propio dispositivo (los sensores RGB) utilizado para la obtención de los nuevos videos.

4.3. Modelos seleccionados

En resumen, el mejor modelo de predicción de frecuencia cardíaca de captura con el dedo resultó en un MAE de 7.58 lpm y un RMSE de 11.08 lpm en la partición de validación. Al calcular estas métricas en el dataset de testeo resultan de 7.02 (MAE) y 10.01 (RMSE), incluso mejores que en la partición de validación, por lo que se comprueba la robustez del modelo. Además el R^2 resulta de 0.84 para el dataset de testeo, por lo que el modelo tiene un buen poder predictivo.

En los desarrollos realizados con videos de la cara, el modelo elegido resultó en un ρ de 0.82 en el entrenamiento con los datos de validación comparando la señal de fotopletismografía original y la señal-predicción. Para poder comparar el modelo con otros modelos previamente entrenados se buscó calcular el RMSE y MAE, los cuales resultaron 4.12 lpm y 2.43 lpm respectivamente en validación. Para probar su robustez con el dataset de UMA-rPPG acotado, los resultados de RMSE y MAE fueron 15,89 y 13,08 lpm como métricas finales.

A pesar de los resultados obtenidos, este modelo presenta una principal ventaja frente a los modelos la cual es la posibilidad de obtener una variedad de parámetros fisiológicos a partir de la señal de FPG. Es por esto que el modelo fue el elegido para ser puesto en producción y realizar iteraciones sobre el mismo para mejorar su rendimiento.

4.4. Desafíos futuros

En primer lugar, los modelos entrenados con el enfoque del dedo cuentan únicamente con la frecuencia cardíaca como etiqueta, en lugar de la señal de fotopletismografía. Como consecuencia, fue necesario asumir que la extracción de la señal de fotopletismografía fue correcta y únicamente se pudo entrenar un modelo que prediga el valor numérico de la frecuencia cardíaca. A futuro, sería útil generar un dataset que contenga la señal de fotopletismografía como etiqueta, permitiendo así desarrollar un modelo capaz de predecir la señal correspondiente a cada video, en lugar de solo la frecuencia cardíaca, tal como se hizo con el enfoque de la cara. Esta mejora resultaría especialmente beneficiosa, ya que la señal de fotopletismografía contiene información valiosa sobre la actividad cardiovascular y su análisis puede proporcionar una mayor comprensión de la salud del individuo.

Al disponer de la señal de fotopletismografía como etiqueta, se abrirían nuevas posibilidades en la monitorización de la salud y el diagnóstico médico. La señal de fotopletismografía no solo brinda información sobre la frecuencia cardíaca, sino que también puede utilizarse para medir parámetros adicionales, como la variabilidad de la frecuencia cardíaca. La variabilidad de la frecuencia cardíaca es un indicador importante del equilibrio autonómico y se ha relacionado con el riesgo de diversas enfermedades cardiovasculares. Por lo tanto, contar con un modelo capaz de predecir la señal de fotopletismografía permitiría obtener mediciones precisas de la variabilidad de la frecuencia cardíaca, lo que contribuiría a una evaluación más completa de la salud cardiovascular de un individuo.

Además, la señal de fotopletismografía también puede utilizarse para identificar anomalías o arritmias cardíacas. La presencia de patrones anormales en la señal puede indicar la existencia de una condición cardíaca anómala, lo que resulta relevante para la detección temprana y el tratamiento oportuno de enfermedades cardiovasculares. Al entrenar un modelo que sea capaz de predecir la señal de fotopletismografía, se podrían implementar algoritmos de detección de anomalías, lo que facilitaría el monitoreo continuo y no invasivo de la salud cardiovascular.

Como instancia final, los modelos entrenados en el enfoque de la cara fueron entrenados con datasets tomados a partir de un solo dispositivo y con condiciones de luz particulares a cada dataset. También se debe considerar la distribución de valores de frecuencia cardíaca en los datasets utilizados para obtener un dataset más balanceado en valores normales y en taquicardia. En los modelos 3DCNN v1.X y v2.X la cantidad de muestras de taquicardia eran menores a las de valores normales. En los modelo tipo 3DCNNv1.X, 66 muestras eran de taquicardia y 17 se encontraban en el dataset de validación; y en los modelos 3DCNNv2.X, únicamente 24 muestras eran de taquicardia, de las cuales 6 se encontraban en el conjunto de validación. El comportamiento de los

valores de predicción en relación a los rangos de valores menores a 77 y mayores a 135 puede solucionarse aumentando la cantidad de datos de forma que la distribución de los valores de la frecuencia cardíaca sea uniforme.

Además, los videos fueron tomados con personas entre los 20 y 50 años de edad, con una gran proporción de personas caucásicas por lo cual no se ha verificado cómo se desempeñan los modelos en niños y adultos mayores, y con pieles más oscuras. Por lo cual, dentro de iteraciones futuras, se buscará obtener videos con diversos dispositivos, condiciones de iluminación, tonos de piel y un amplio rango de edades para poder testear la robustez del algoritmo con una mayor variedad de condiciones. Una vez obtenidos esos datos, se procederá a reentrenar el modelo con *output* de fotopletismografía. Dado que este modelo fue entrenado con videos únicamente preprocesados con detección de rostro, se analizará integrar al preprocesamiento la magnificación euleriana para mejorar su rendimiento. En futuras iteraciones, además, se propone investigar cómo repercuten otros algoritmos de segmentación del área de interés y magnificación de la señal en el desempeño del modelo elegido para la extracción de las fotopletismografías.

Por último, la ventaja significativa del entrenamiento con la función de correlación como función de pérdida es el buen posicionamiento de picos. Esto nos permitirá calcular la variabilidad de frecuencia cardíaca con gran exactitud [32] y poder detectar casos de arritmias, entre otras patologías. En conjunto con la predicción de la señal de fotopletismografía, se podrán agregar dichos algoritmos al flujo e incluir la estimación de otros parámetros fisiológicos (frecuencia respiratoria, presión arterial, marcadores de estrés y fatiga, entre otros) de manera sincronizada para un exámen médico completo de manera remota.

Conclusiones

Durante este trabajo se ha demostrado que la estimación de la frecuencia cardíaca mediante dispositivos móviles es factible, utilizando algoritmos de inteligencia artificial basados en Deep Learning, tanto de forma remota como por contacto.

En primer lugar, se logró entrenar un modelo de predicción de frecuencia cardíaca mediante videos del dedo índice que no solo se adapta a condiciones controladas de laboratorio sino también a capturas con otros dispositivos y distintas condiciones lo cual resulta prometedor como una herramienta versátil y aplicable al mundo real. En el enfoque de fotopletismografía remota, se encontraron signos favorables de generalización y robustez con datasets limitados, y si bien su performance es menor al método por contacto, sus ventajas la posicionan con un potencial enorme.

Durante el proceso de investigación y desarrollo se destaca dos instancias de descubrimiento claves que se consideran de utilidad para investigaciones futuras:

1. La implementación de un modelo de Deep Learning basado en imágenes de las señales presentes en los canales de RGB que clasifique la calidad del input previo a la entrada al modelo, el cual terminó siendo un factor decisivo para explicar el rendimiento y la robustez de los modelos de fotopletismografía de contacto.
2. La posibilidad de predecir no solo la frecuencia cardíaca sino haberla obtenido de forma indirecta a partir de la estimación de la onda de pulso presenta ventajas en la información que esta contiene, y se cree posiblemente que podrán ser estimadas a futuro otros parámetros como resistencia vascular, volumen minuto o detección de patología aórtica.

La telemedicina y la inteligencia artificial aplicada a la salud han demostrado grandes avances en los últimos años con el objetivo de solucionar problemáticas clave de la situación mundial. La accesibilidad a los servicios y el costo de los mismos son parte de los temas que se busca tratar con este proyecto al brindar una herramienta a través de dispositivos comerciales, comúnmente ya utilizados en la vida diaria, que son capaces de extraer la fotopletismografía y la frecuencia cardíaca. Estos parámetros fisiológicos son indicadores de salud esenciales para conocer el estado de un paciente y con una extracción remota, como la que se plantea en este trabajo, es posible acceder a dichos

datos de forma no invasiva y frente a escenarios que imposibiliten un examen físico completo tales como el aislamiento por pandemia, imposibilidades de traslado por comorbilidad, movilidad disminuida o imposibilidad de usos prolongados de los sensores lumínicos por sensibilidad en la piel y/o preferencias del usuario.

La posibilidad de obtener estos parámetros de forma remota a través de un dispositivo móvil como un smartphone, convierten la solución planteada en un herramienta que contribuiría a brindar atención médica de calidad de una forma accesible, no sólo brindando información a los profesionales médicos, sino involucrando al paciente en un control periódico de su propia salud,

Apéndice A

Tablas y figuras anéxas

A.1. Estructuras de redes convolucionales para los modelos de fotopletismografía de contacto

Nº Bloque	Capa	Formato de salida	Cantidad de parámetros
1	Conv1D	(None, 240, 64)	3904
1	MaxPooling1D	(None, 120, 64)	0
2	Conv1D	(None, 91, 30)	57630
2	MaxPooling1D	(None, 45, 30)	0
3	conv1d_5 (Conv1D)	(None, 16, 20)	18020
3	MaxPooling1D	(None, 8, 20)	0
-	Flatten	(None, 160)	0
-	Dense	(None, 1)	161

Tabla A.1: Estructura de la red convolucional 1D correspondiente al modelo 1DCNNv2.1

Nº Bloque	Capa	Formato de salida	Cantidad de parámetros
1	Conv1D	(None, 297, 10)	40
1	MaxPooling1D	(None, 148, 10)	0
2	Conv1D	(None, 146, 10)	310
2	MaxPooling1D	(None, 73, 10)	0
3	Conv1D	(None, 71, 10)	310
3	MaxPooling1D	(None, 35, 10)	0
4	Conv1D	(None, 33, 10)	310
4	MaxPooling1D	(None, 16, 10)	0
5	Conv1D	(None, 14, 10)	310
5	MaxPooling1D	(None, 7, 10)	0
-	Flatten	(None, 70)	0
-	Dense	(None, 1)	71

Tabla A.2: Estructura de la red convolucional 1D correspondiente al modelo 1DCNNv2.2

Nº Bloque	Capa	Formato de salida	Cantidad de parámetros
1	Conv1D	(None, 295, 30)	180
1	MaxPooling1D	(None, 147, 30)	0
2	Conv1D	(None, 143, 30)	4530
2	MaxPooling1D	(None, 71, 30)	0
3	Conv1D	(None, 67, 30)	4530
3	MaxPooling1D	(None, 33, 30)	0
4	Conv1D	(None, 29, 30)	4530
4	MaxPooling1D	(None, 14, 30)	0
5	Conv1D	(None, 10, 30)	4530
5	MaxPooling1D	(None, 5, 30)	0
-	Flatten	(None, 150)	0
-	Dense	(None, 1)	151

Tabla A.3: Estructura de la red convolucional 1D correspondiente a los modelos 1DCNNv2.3 y 1DCNNv2.6

Nº Bloque	Capa	Formato de salida	Cantidad de parámetros
1	Conv1D	(None, 293, 50)	400
1	Dropout	(None, 293, 50)	0
1	MaxPooling1D	(None, 146, 50)	0
2	Conv1D	(None, 140, 50)	17550
2	Dropout	(None, 140, 50)	0
2	MaxPooling1D	(None, 70, 50)	0
3	Conv1D	(None, 64, 50)	17550
3	Dropout	(None, 64, 50)	0
3	MaxPooling1D	(None, 32, 50)	0
4	Conv1D	(None, 26, 50)	17550
4	Dropout	(None, 26, 50)	0
4	MaxPooling1D	(None, 13, 50)	0
5	Conv1D	(None, 7, 50)	17550
5	MaxPooling1D	(None, 3, 50)	0
-	Flatten	(None, 150)	0
-	Dense	(None, 1)	151

Tabla A.4: Estructura de la red convolucional 1D correspondiente al modelo 1DCNNv2.6

A.2. Tablas de resultados para los modelos de fotopletismografía remota con output de valor escalar

Modelo	Dataset	MAE (lpm)	RMSE (lpm)
<i>3DCNNv1.0</i>	<i>Entrenamiento</i>	2.772	3.576
<i>3DCNNv1.1</i>	<i>Entrenamiento</i>	5.2285	7.2566
<i>3DCNNv1.2</i>	<i>Entrenamiento</i>	3.2610	4.1689
<i>3DCNNv1.3</i>	<i>Entrenamiento</i>	5.3282	7.0244
<i>3DCNNv2.0</i>	<i>Entrenamiento</i>	2.639	3.289
<i>3DCNNv2.1</i>	<i>Entrenamiento</i>	4.0851	5.1985
<i>3DCNNv2.2</i>	<i>Entrenamiento</i>	8.4398	10.9502
<i>3DCNNv2.3</i>	<i>Entrenamiento</i>	1.7667	2.1920
<i>3DCNNv2.4</i>	<i>Entrenamiento</i>	2.7819	3.7212

Tabla A.5: Métricas de los modelos en entrenamiento con *output* de frecuencia cardíaca: MAE y RMSE.

Modelo	Dataset	MAE (lpm)	RMSE (lpm)	ρ	p-value
<i>3DCNNv1.0</i>	<i>Validación</i>	9.496	13.562	0.875	4.98e-17
<i>3DCNNv1.1</i>	<i>Validación</i>	8.3649	11.4874	0.8903	7.42e-14
<i>3DCNNv1.2</i>	<i>Validación</i>	13.02137	17.2988	0.8272	1.53e-10
<i>3DCNNv1.3</i>	<i>Validación</i>	15.5091	20.4235	0.8366	6.048e-11
<i>3DCNNv2.0</i>	<i>Validación</i>	7.725	9.633	0.925	7.44e-16
<i>3DCNNv2.1</i>	<i>Validación</i>	8.4540	10.4539	0.9157	5.11e-15
<i>3DCNNv2.2</i>	<i>Validación</i>	8.1991	10.1885	0.9166	4.24e-15
<i>3DCNNv2.3</i>	<i>Validación</i>	8.7733	11.7172	0.9098	1.54e-14
<i>3DCNNv2.4</i>	<i>Validación</i>	9.4971	11.9482	0.8722	4.27e-12

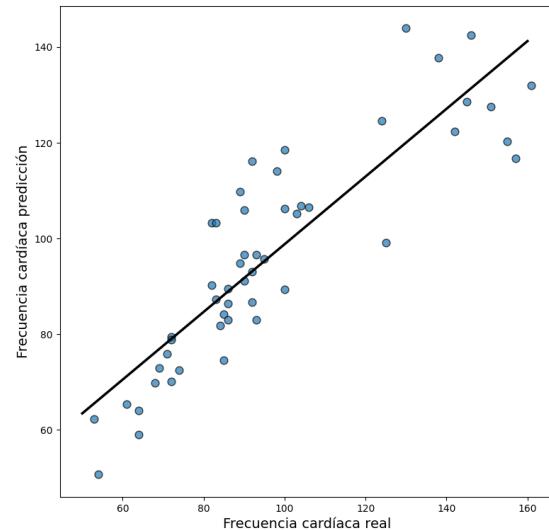
Tabla A.6: Métricas de los modelos en validación con *output* de frecuencia cardíaca: MAE, RMSE y correlación entre la frecuencia cardíaca predicha y la frecuencia medida por el oxímetro.

Modelo	Dataset	MAE (lpm)	RMSE (lpm)	ρ	p-value
<i>3DCNNv1.0</i>	<i>Test</i>	52.4121	55.8698	-	-
<i>3DCNNv1.0</i>	<i>Test acotado</i>	35.2269	38.2722	-0.1055	0.1267
<i>3DCNNv1.1</i>	<i>Test</i>	34.3924	39.1429	-	-
<i>3DCNNv1.1</i>	<i>Test acotado</i>	20.6252	25.0665	0.10609	0.1245
<i>3DCNNv1.2</i>	<i>Test</i>	51.9396	55.3333	-	-
<i>3DCNNv1.2</i>	<i>Test acotado</i>	34.4675	36.5689	-0.00811	0.9067
<i>3DCNNv1.3</i>	<i>Test</i>	33.2291	36.8576	-	-
<i>3DCNNv1.3</i>	<i>Test acotado</i>	18.3018	20.5675	0.08145	0.2388
<i>3DCNNv2.0</i>	<i>Test</i>	20.895	24.244	-	-
<i>3DCNNv2.0</i>	<i>Test acotado</i>	11.704	15.172	0.1937	0.00528
<i>3DCNNv2.1</i>	<i>Test</i>	28.0045	33.7273	-	-
<i>3DCNNv2.1</i>	<i>Test acotado</i>	15.4212	19.8733	-0.2624	0.00001391
<i>3DCNNv2.2</i>	<i>Test</i>	25.4103	30.6224	-	-
<i>3DCNNv2.2</i>	<i>Test acotado</i>	14.7303	19.0578	-0.3447	3.88e-07
<i>3DCNNv2.3</i>	<i>Test</i>	35.0587	39.8008	-	-
<i>3DCNNv2.3</i>	<i>Test acotado</i>	17.5232	20.7706	-0.1458	0.0365
<i>3DCNNv2.4</i>	<i>Test</i>	23.2273	26.9329	-	-
<i>3DCNNv2.4</i>	<i>Test acotado</i>	11.7351	13.8541	0.0087	0.9010

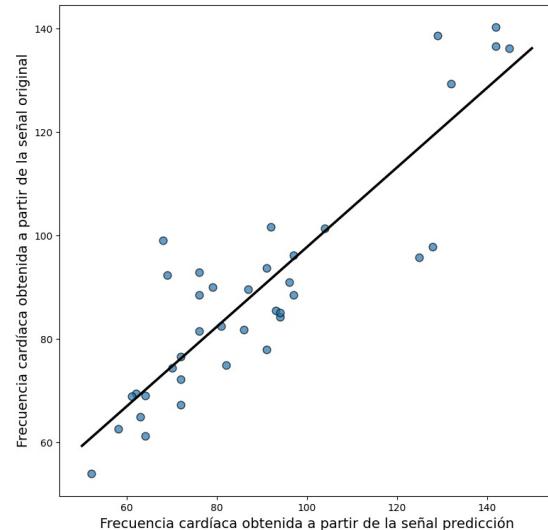
Tabla A.7: Métricas de los modelos en Test y Test acotado con *output* de frecuencia cardíaca: MAE, RMSE y correlación entre la frecuencia cardíaca predicha y la frecuencia medida por el oxímetro.

A.3. Gráficos de resultados

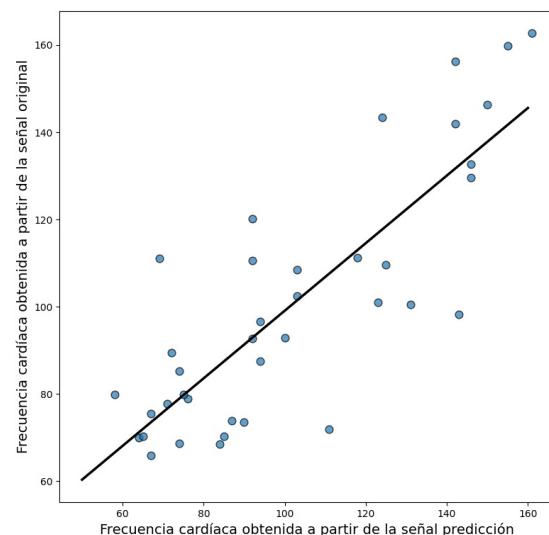
Gráficos para los modelos de FPG remota con output de valor escalar en validación



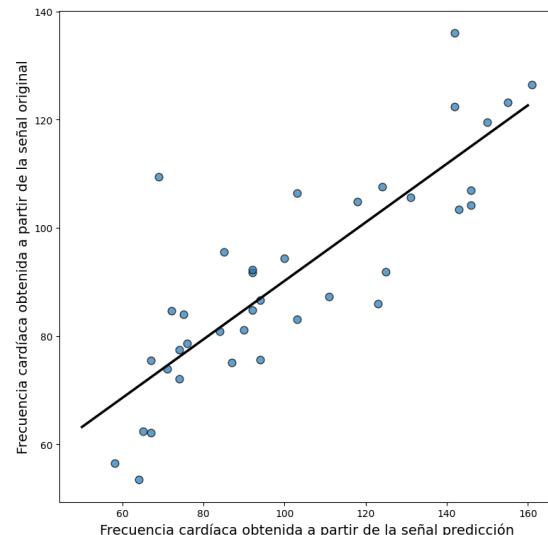
(a) Modelo 3DCNNv1.0



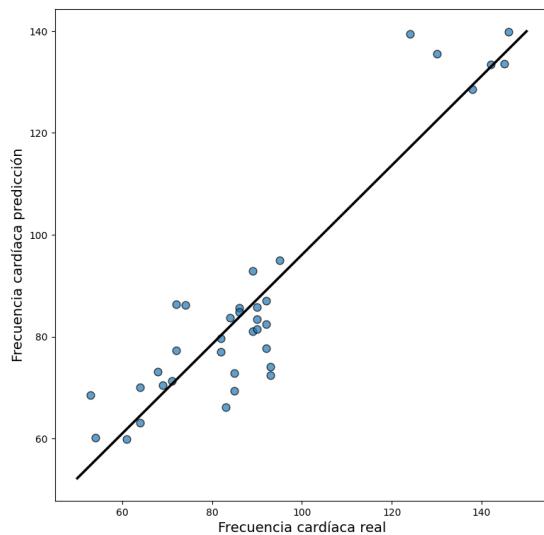
(b) Modelo 3DCNNv1.1



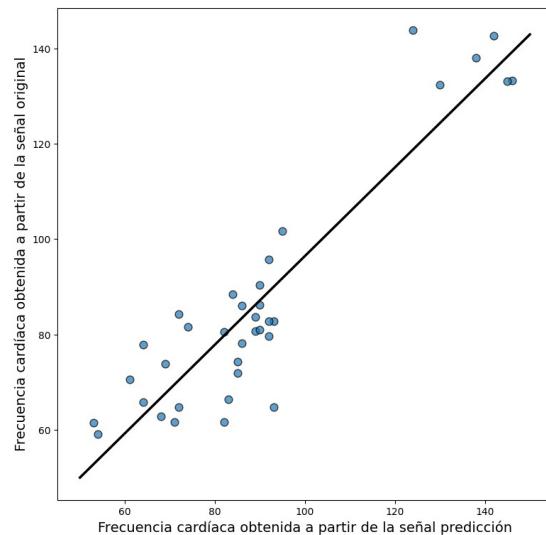
(c) Modelo 3DCNNv1.2



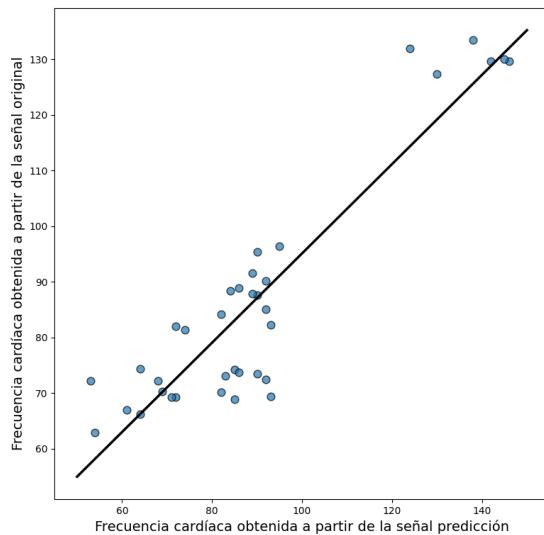
(d) Modelo 3DCNNv1.3



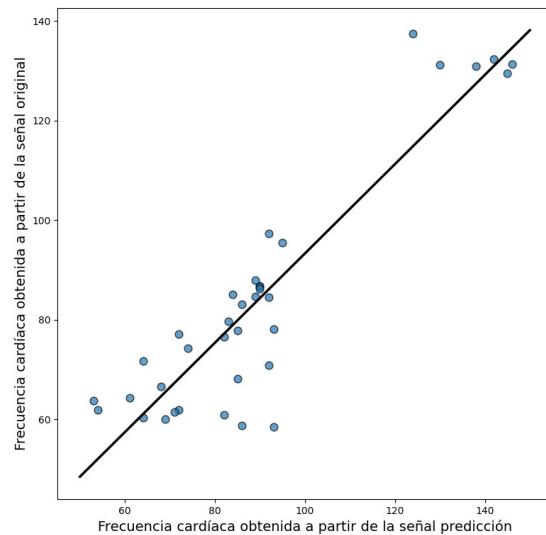
(e) Modelo 3DCNNv2.0



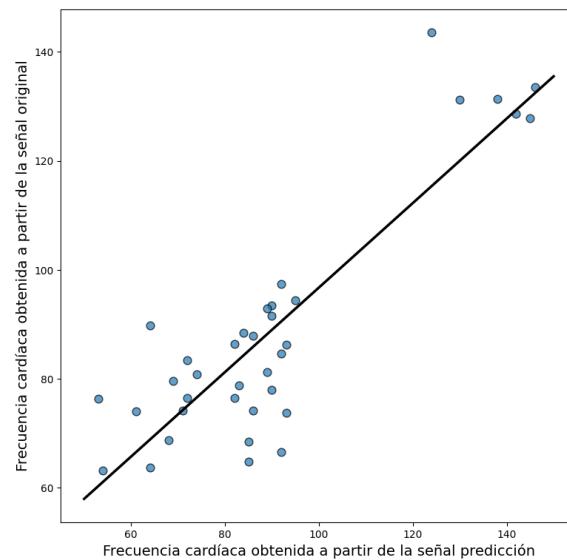
(f) Modelo 3DCNNv2.1



(g) Modelo 3DCNNv2.2



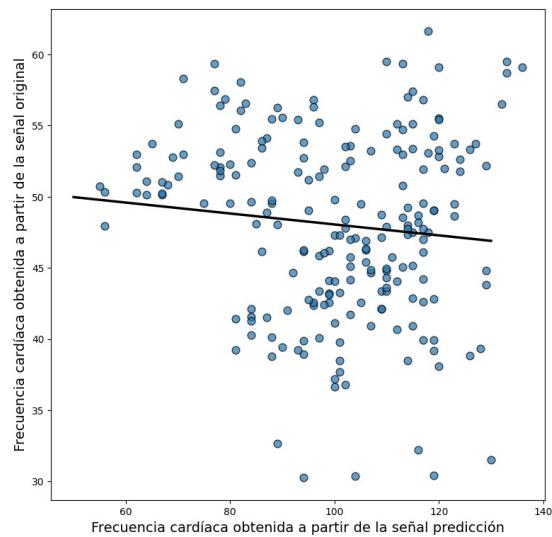
(h) Modelo 3DCNNv2.3



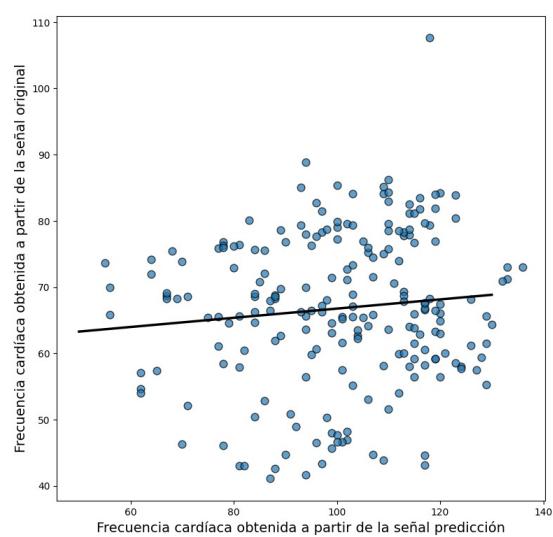
(i) Modelo 3DCNNv2.4

Figura A.1: Frecuencias cardíacas predichas por los modelos con *output* valor escalar en función de las frecuencias cardíacas obtenidas del oxímetro en el dataset de validación.

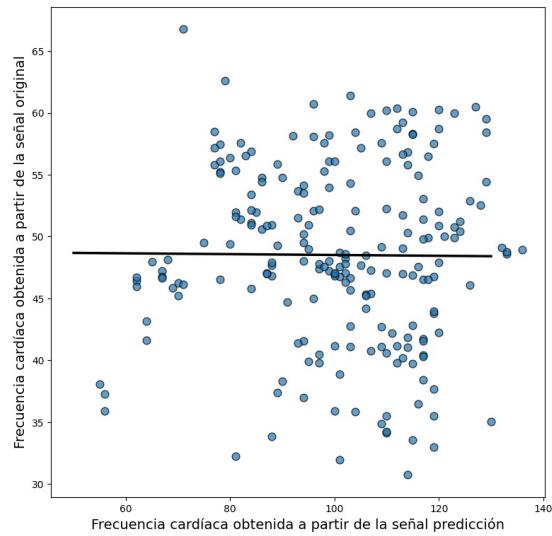
Gráficos para los modelos de FPG remota con output de valor escalar en testeo



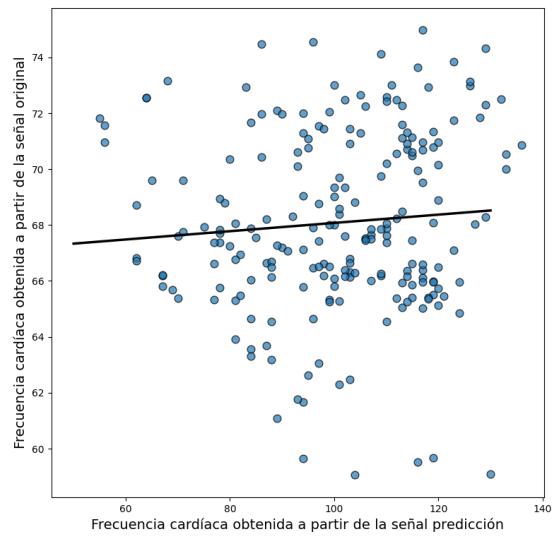
(a) Modelo 3DCNNv1.0



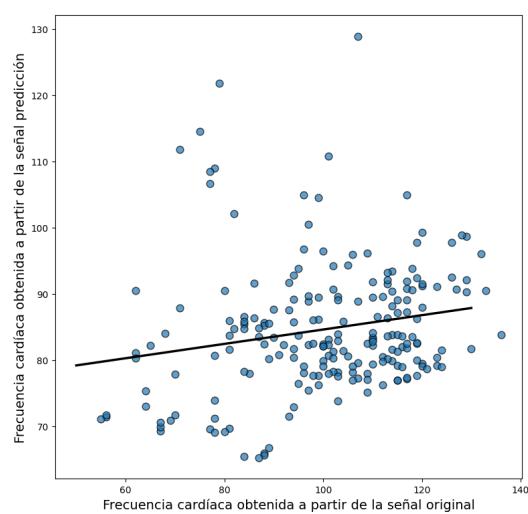
(b) Modelo 3DCNNv1.1



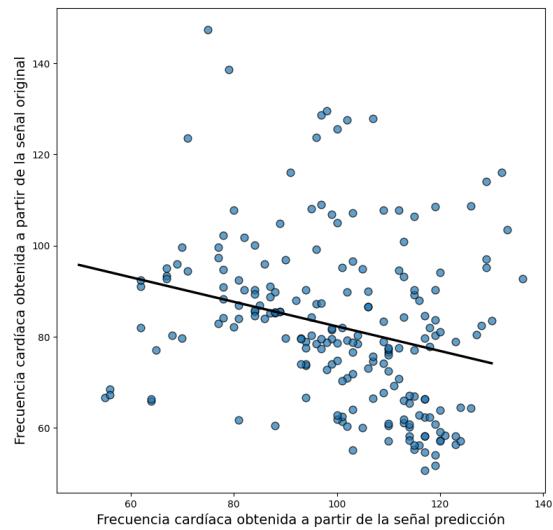
(c) Modelo 3DCNNv1.2



(d) Modelo 3DCNNv1.3



(e) Modelo 3DCNNv2.0



(f) Modelo 3DCNNv2.1

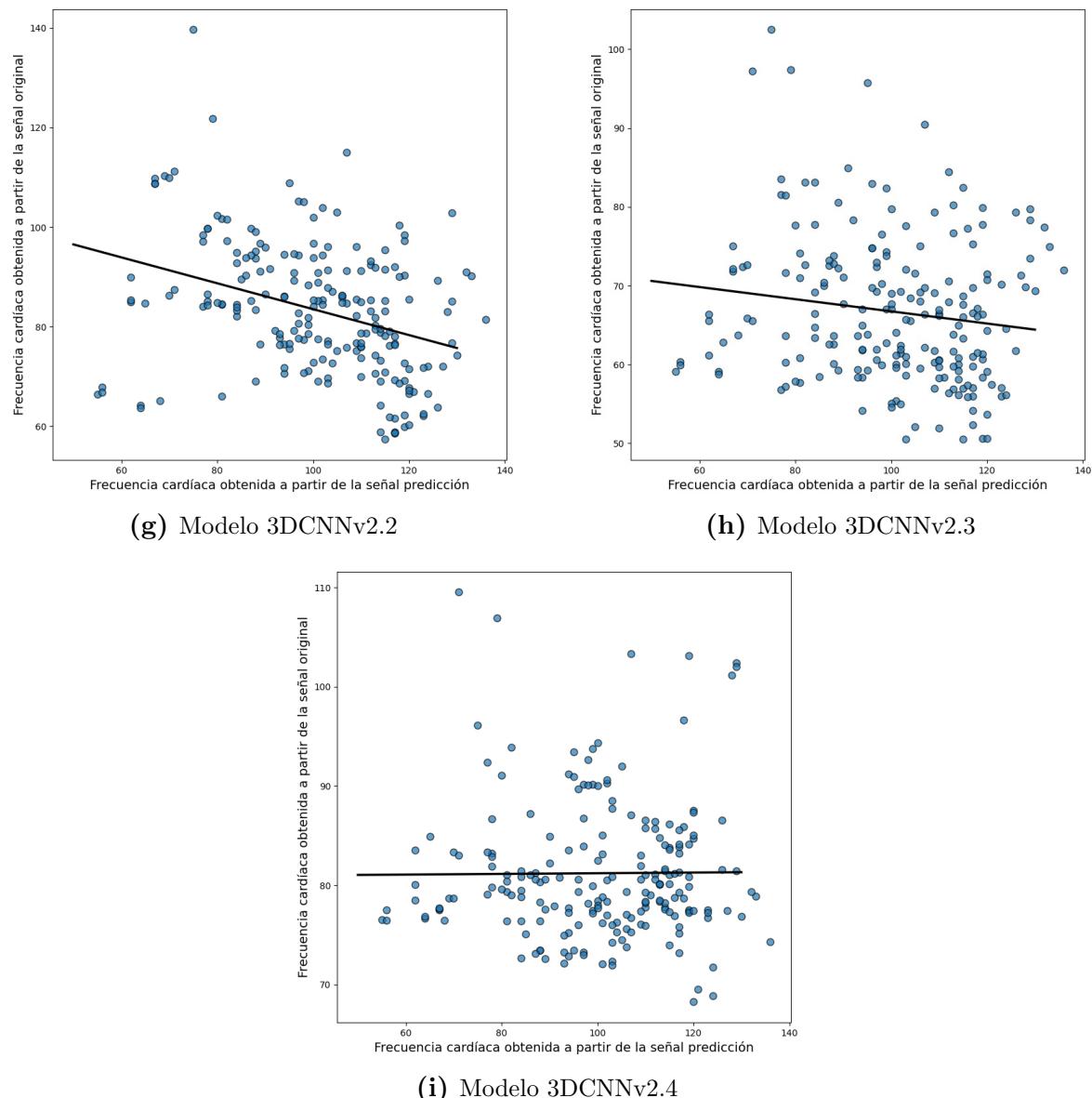
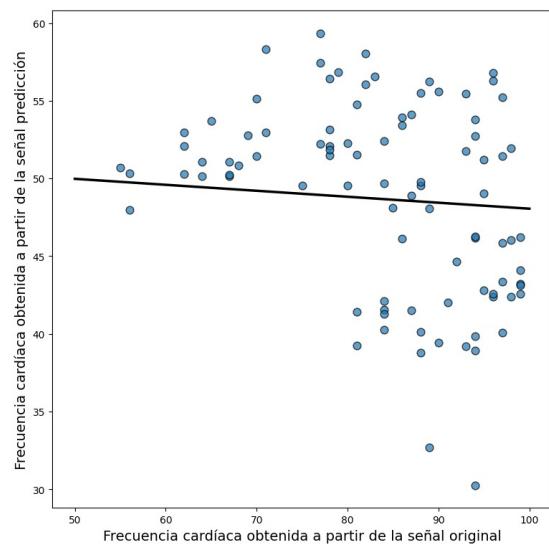
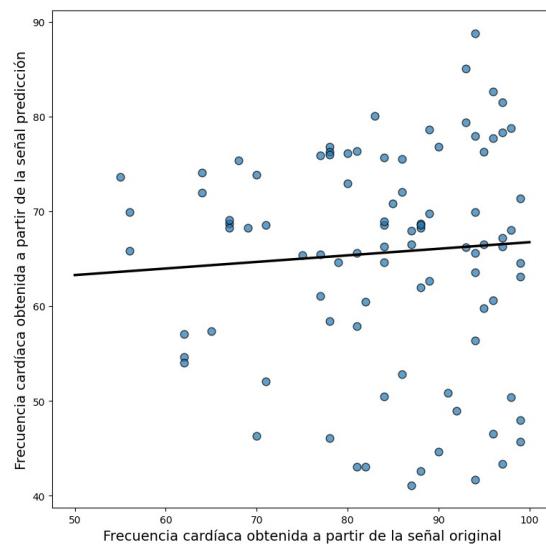


Figura A.2: Frecuencias cardíacas predichas por los modelos con *output* valor escalar en función de las frecuencias cardíacas obtenidas del oxímetro en el dataset de testeo.

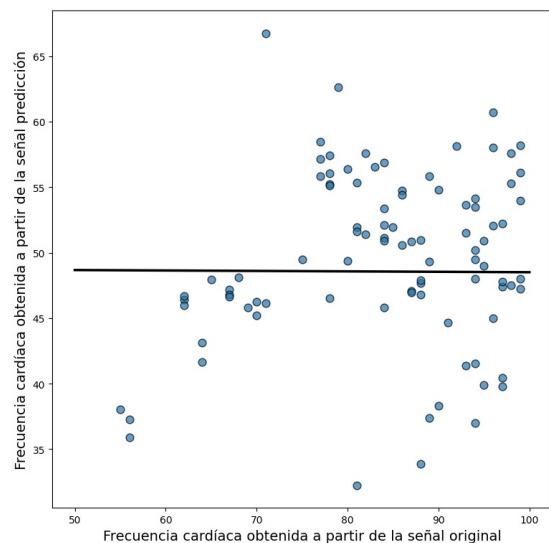
Gráficos para los modelos de FPG remota con output de valor escalar en testeo acotado



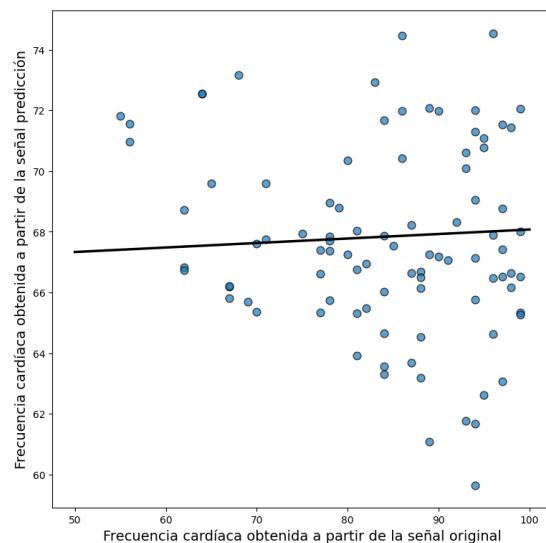
(a) Modelo 3DCNNv1.0



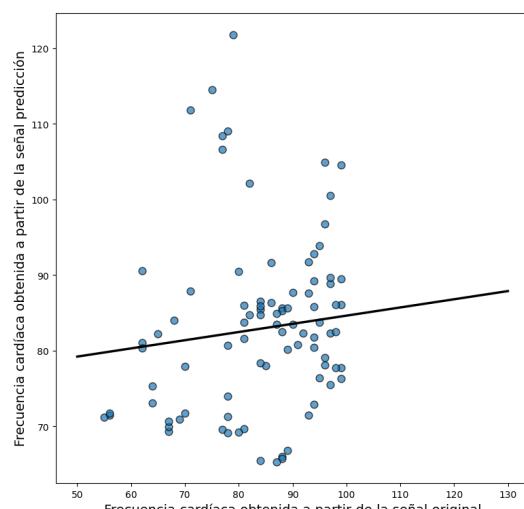
(b) Modelo 3DCNNv1.1



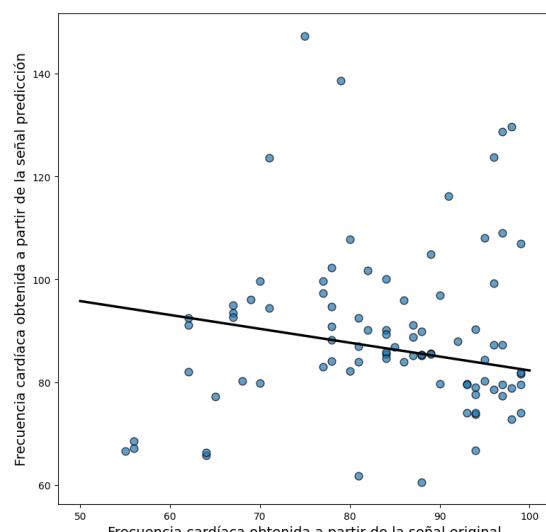
(c) Modelo 3DCNNv1.2



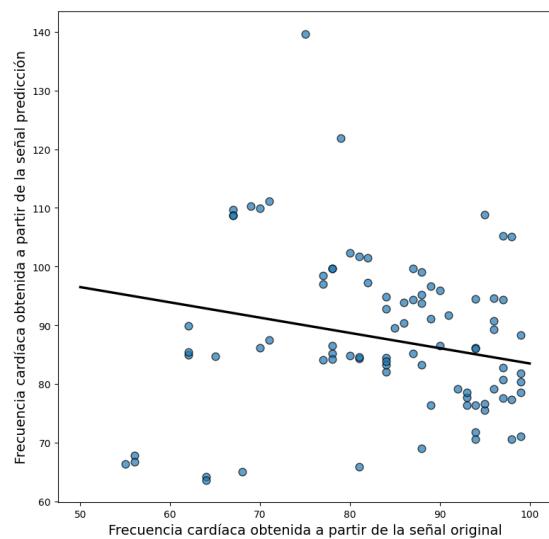
(d) Modelo 3DCNNv1.3



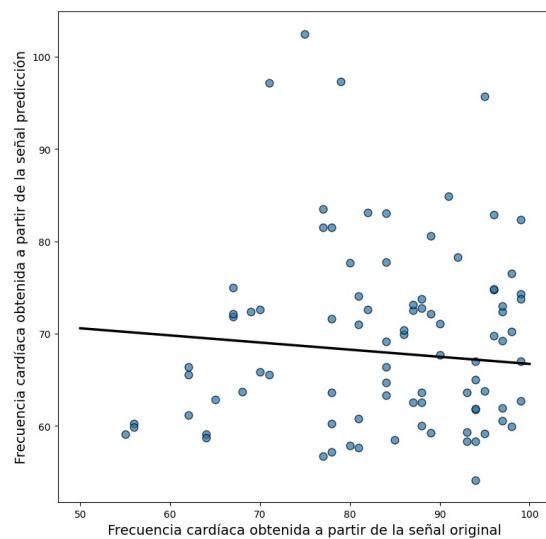
(e) Modelo 3DCNNv2.0



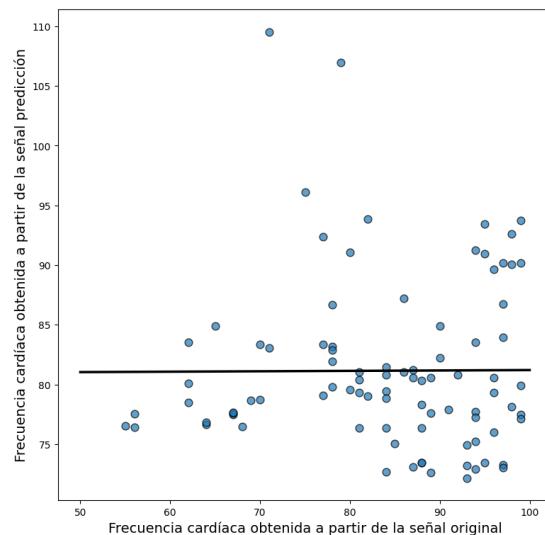
(f) Modelo 3DCNNv2.1



(g) Modelo 3DCNNv2.2



(h) Modelo 3DCNNv2.3



(i) Modelo 3DCNNv2.4

Figura A.3: Frecuencias cardíacas predichas por los modelos con *output* valor escalar en función de las frecuencias cardíacas obtenidas del oxímetro en el dataset de testeo limitado a valores menores a 100 lpm.

Bibliografía

- [1] World health statistics 2022: monitoring health for the SDGs, sustainable development goals - Pathway to Universal Health Coverage. págs. 60 –72, 2022. URL <https://www.who.int/publications/i/item/9789240051157>. v, 1, 2
- [2] Acceso y uso de tecnologías de la información y la comunicación. encuesta permanente de hogares. *Informes Técnicos: Ciencia y Tecnología*, **6** (89), 2021. URL https://www.indec.gob.ar/uploads/informesdeprensa/mautic_05_22843D61C141.pdf. v, 3
- [3] Accesos a internet. encuesta permanente de hogares. *Informes Técnicos: Servicios*, **6** (165), 2021. URL https://www.indec.gob.ar/uploads/informesdeprensa/internet_09_2260723E2261.pdf. v, 3, 4
- [4] Tamura, T. Current progress of photoplethysmography and spo2 for health monitoring. *Biomedical engineering letters*, **9** (1), 21–36, 2019. v, v, 9, 10
- [5] Rana, A., Singh Rawat, A., Bijalwan, A., Bahuguna, H. Application of multi layer (perceptron) artificial neural network in the diagnosis system: A systematic review. En: 2018 International Conference on Research in Intelligent and Computing in Engineering (RICE), págs. 1–6. 2018. v, 15, 17
- [6] Multi-layer perceptron vs. deep neural network, 2022. URL <https://www.baeldung.com/cs/mlp-vs-dnn#:~:text=MLPs%20are%20neural%20networks%20with,to%20traditional%20machine%20learning%20algorithms>. v, 19, 20
- [7] Kozlowski, D., Barriola, J. Clase 12. redes neuronales ii, 2018. URL <https://diegokoz.github.io/EEA2019/clase%2012/mnist102.nb.html#fn1>. v, v, 23, 26, 27
- [8] Dharmaraj. Zero-padding in convolutional neural networks, 2021. URL <https://medium.com/@draj0718/zero-padding-in-convolutional-neural-networks-bf1410438e99>. v, 24

- [9] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., *et al.* Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions. *Journal of big Data*, **8** (1), 1–74, 2021. [v](#), [14](#), [18](#), [20](#), [21](#), [24](#), [25](#), [26](#), [28](#)
- [10] Tensorflow. Video classification with a 3d convolutional neural network (cnn), 2022. URL https://www.tensorflow.org/tutorials/video/video_classification. [vi](#), [30](#)
- [11] Bae, S., Borac, S., Emre, Y., Wang, J., Wu, J., Kashyap, M., *et al.* Prospective validation of smartphone-based heart rate and respiratory rate measurement algorithms. *Communications medicine*, **2** (1), 40, 2022. [vi](#), [vi](#), [37](#)
- [12] Wang, Y.-Q. An analysis of the viola-jones face detection algorithm. *Image Processing On Line*, **4**, 128–148, 2014. [vi](#), [58](#)
- [13] What is face detection? ultimate guide 2023 + model comparison, 2023. URL <https://learnopencv.com/what-is-face-detection-the-ultimate-guide/#Classical-Algorithms-of-Face-Detection>. [vi](#), [58](#), [60](#)
- [14] Jaiswal, K. B., Meenpal, T. Heart rate estimation network from facial videos using spatiotemporal feature image. *Computers in Biology and Medicine*, **151**, 106307, 2022. [viii](#), [7](#)
- [15] La declaración universal de derechos humanos, 2022. URL <https://www.un.org/es/about-us/universal-declaration-of-human-rights>. [1](#)
- [16] Cobertura universal de la salud - español, 2022. URL [https://www.who.int/es/news-room/fact-sheets/detail/universal-health-coverage-\(uhc\)](https://www.who.int/es/news-room/fact-sheets/detail/universal-health-coverage-(uhc)). [1](#)
- [17] Ageing, 2020. URL https://www.who.int/health-topics/ageing#tab=tab_1. [2](#)
- [18] Universal health coverage - english, 2021. URL [https://www.who.int/es/news-room/fact-sheets/detail/universal-health-coverage-\(uhc\)](https://www.who.int/es/news-room/fact-sheets/detail/universal-health-coverage-(uhc)). [2](#)
- [19] Universal health coverage, 2021. URL [https://www.who.int/news-room/fact-sheets/detail/universal-health-coverage-\(uhc\)](https://www.who.int/news-room/fact-sheets/detail/universal-health-coverage-(uhc)). [2](#)
- [20] Verkruysse, W., Svaasand, L. O., Nelson, J. S. Remote plethysmographic imaging using ambient light. *Opt. Express*, **16** (26), 21434–21445, Dec 2008. URL <http://opg.optica.org/oe/abstract.cfm?URI=oe-16-26-21434>. [3](#)
- [21] Khan, R. S., Zardar, A. A., Bhatti, Z. Artificial intelligence based smart doctor using decision tree algorithm. *arXiv preprint arXiv:1808.01884*, 2018.

- [22] Bisepro, el proyecto que detectará la sepsis con tecnología big data, 2021. URL <https://www.iic.uam.es/noticias/iic-presentacion-proyecto-bisepro/>.
- [23] Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., *et al.* Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *arXiv preprint arXiv:1711.05225*, 2017. 3
- [24] Cheng, C.-H., Wong, K.-L., Chin, J.-W., Chan, T.-T., So, R. H. Deep learning methods for remote heart rate measurement: a review and future research agenda. *Sensors*, **21** (18), 6296, 2021. 5
- [25] Poh, M.-Z., McDuff, D. J., Picard, R. W. Non-contact, automated cardiac pulse measurements using video imaging and blind source separation. *Optics express*, **18** (10), 10762–10774, 2010. 5, 7
- [26] Lewandowska, M., Rumiński, J., Kocejko, T., Nowak, J. Measuring pulse rate with a webcam—a non-contact method for evaluating cardiac activity. En: 2011 federated conference on computer science and information systems (FedCSIS), págs. 405–410. IEEE, 2011. 5
- [27] De Haan, G., Jeanne, V. Robust pulse rate from chrominance-based rppg. *IEEE Transactions on Biomedical Engineering*, **60** (10), 2878–2886, 2013. 5, 7
- [28] Wang, W., Stuijk, S., De Haan, G. A novel algorithm for remote photoplethysmography: Spatial subspace rotation. *IEEE transactions on biomedical engineering*, **63** (9), 1974–1984, 2015. 5
- [29] Wang, W., Den Brinker, A. C., Stuijk, S., De Haan, G. Algorithmic principles of remote ppg. *IEEE Transactions on Biomedical Engineering*, **64** (7), 1479–1491, 2016. 5
- [30] Chen, W., McDuff, D. Deepphys: Video-based physiological measurement using convolutional attention networks. En: Proceedings of the european conference on computer vision (ECCV), págs. 349–365. 2018. 5, 7
- [31] Špetlík, R., Franc, V., Matas, J. Visual heart rate estimation with convolutional neural network. En: Proceedings of the british machine vision conference, Newcastle, UK, págs. 3–6. 2018. 5, 7
- [32] Yu, Z., Li, X., Zhao, G. Remote photoplethysmograph signal measurement from facial videos using spatio-temporal networks. *arXiv preprint arXiv:1905.02419*, 2019. 5, 6, 7, 57, 70, 86

- [33] Yu, Z., Peng, W., Li, X., Hong, X., Zhao, G. Remote heart rate measurement from highly compressed facial videos: an end-to-end deep learning solution with video enhancement. En: Proceedings of the IEEE/CVF International Conference on Computer Vision, págs. 151–160. 2019. [5](#), [61](#), [64](#), [65](#), [67](#)
- [34] Yu, Z., Li, X., Niu, X., Shi, J., Zhao, G. Autohr: A strong end-to-end baseline for remote heart rate measurement with neural searching. *IEEE Signal Processing Letters*, **27**, 1245–1249, 2020. [5](#), [7](#)
- [35] Hu, M., Qian, F., Guo, D., Wang, X., He, L., Ren, F. Eta-rppgnet: effective time-domain attention network for remote heart rate measurement. *IEEE Transactions on Instrumentation and Measurement*, **70**, 1–12, 2021. [5](#)
- [36] Hu, M., Qian, F., Wang, X., He, L., Guo, D., Ren, F. Robust heart rate estimation with spatial-temporal attention network from facial videos. *IEEE Transactions on Cognitive and Developmental Systems*, 2021.
- [37] Zhang, P., Li, B., Peng, J., Jiang, W. Multi-hierarchical convolutional network for efficient remote photoplethysmograph signal and heart rate estimation from face video clips. *arXiv preprint arXiv:2104.02260*, 2021. [5](#)
- [38] Sabokrou, M., Pourreza, M., Li, X., Fathy, M., Zhao, G. Deep-hr: Fast heart rate estimation from face video under realistic conditions. *Expert Systems with Applications*, **186**, 115596, 2021. [6](#)
- [39] Tang, C., Lu, J., Liu, J. Non-contact heart rate monitoring by combining convolutional neural network skin detection and remote photoplethysmography via a low-cost camera. En: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, págs. 1309–1315. 2018. [6](#)
- [40] Chen, C. H. Handbook of pattern recognition and computer vision. World Scientific, 2015. [6](#)
- [41] Botina-Monsalve, D., Benezeth, Y., Macwan, R., Pierrart, P., Parra, F., Nakamura, K., *et al.* Long short-term memory deep-filter in remote photoplethysmography. En: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, págs. 306–307. 2020. [6](#)
- [42] Liu, X., Jiang, Z., Fromm, J., Xu, X., Patel, S., McDuff, D. Metaphys: few-shot adaptation for non-contact physiological measurement. En: Proceedings of the conference on health, inference, and learning, págs. 154–163. 2021. [6](#)

- [43] Liu, S.-Q., Yuen, P. C. A general remote photoplethysmography estimator with spatiotemporal convolutional network. En: 2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020), págs. 481–488. IEEE, 2020. [6](#)
- [44] Perepelkina, O., Artemyev, M., Churikova, M., Grinenko, M. Hearttrack: Convolutional neural network for remote video-based heart rate monitoring. En: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, págs. 288–289. 2020.
- [45] Tsou, Y.-Y., Lee, Y.-A., Hsu, C.-T., Chang, S.-H. Siamese-rppg network: remote photoplethysmography signal estimation from face videos. *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, 2020. [6](#)
- [46] Huang, B., Lin, C.-L., Chen, W., Juang, C.-F., Wu, X. A novel one-stage framework for visual pulse rate estimation using deep neural networks. *Biomedical Signal Processing and Control*, **66**, 102387, 2021. [6](#)
- [47] Perepelkina, O., Artemyev, M., Churikova, M., Grinenko, M. Hearttrack: Convolutional neural network for remote video-based heart rate monitoring. En: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. 2020. [6](#)
- [48] He, K., Zhang, X., Ren, S., Sun, J. Deep residual learning for image recognition. En: Proceedings of the IEEE conference on computer vision and pattern recognition, págs. 770–778. 2016. [6](#)
- [49] Lu, H., Han, H. Nas-hr: Neural architecture search for heart rate estimation from face videos. *Virtual Reality & Intelligent Hardware*, **3** (1), 33–42, 2021. [6](#)
- [50] Song, R., Zhang, S., Li, C., Zhang, Y., Cheng, J., Chen, X. Heart rate estimation from facial videos using a spatiotemporal representation with convolutional neural networks. *IEEE Transactions on Instrumentation and Measurement*, **69** (10), 7411–7421, 2020. [6](#), [7](#)
- [51] Jonathan, E., Leahy, M. Investigating a smartphone imaging unit for photoplethysmography. *Physiological measurement*, **31** (11), N79, 2010. [6](#)
- [52] Siddiqui, S. A., Zhang, Y., Feng, Z., Kos, A. A pulse rate estimation algorithm using ppg and smartphone camera. *Journal of medical systems*, **40**, 1–6, 2016. [6](#), [7](#)

- [53] Yan, B. P., Chan, C. K., Li, C. K., To, O. T., Lai, W. H., Tse, G., *et al.* Resting and postexercise heart rate detection from fingertip and facial photoplethysmography using a smartphone camera: a validation study. *JMIR mHealth and uHealth*, **5** (3), e7275, 2017. [6](#), [7](#)
- [54] Sun, Y., Thakor, N. Photoplethysmography revisited: From contact to noncontact, from point to imaging. *IEEE Transactions on Biomedical Engineering*, **63** (3), 463–477, 2016. [8](#), [9](#)
- [55] Meada, Y., Sekine, M., Tamura, T. The advantage of green reflected photoplethysmograph. *J. Med. Syst.*, **35**, 829–834, 2011. [9](#)
- [56] Maeda, Y., Sekine, M., Tamura, T., Moriya, A., Suzuki, T., Kameyama, K. Comparison of reflected green light and infrared photoplethysmography. págs. 2270–2272, 2008. [9](#)
- [57] Lee, J., Matsumura, K., Yamakoshi, K.-i., Rolfe, P., Tanaka, S., Yamakoshi, T. Comparison between red, green and blue light reflection photoplethysmography for heart rate monitoring during motion. págs. 1724–1727, 2013. [9](#)
- [58] ¿qué es la inteligencia artificial?, 2022. URL <https://www.ibm.com/ar-es/cloud/learn/what-is-artificial-intelligence>. [12](#), [13](#), [14](#)
- [59] Secinaro, S., Calandra, D., Secinaro, A., Muthurangu, V., Biancone, P. The role of artificial intelligence in healthcare: a structured literature review. *BMC Medical Informatics and Decision Making*, **21** (1), 1–23, 2021. [12](#)
- [60] Supervised vs. unsupervised learning: What's the difference?, 2022. URL <https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning>. [13](#)
- [61] Supervised learning, 2022. URL <https://www.ibm.com/cloud/learn/supervised-learning>. [13](#)
- [62] Unsupervised learning, 2022. URL <https://www.ibm.com/cloud/learn/unsupervised-learning>. [13](#)
- [63] Transfer learning para deep learning, 2022. URL <https://developer.ibm.com/articles/transfer-learning-for-deep-learning/>. [14](#)
- [64] Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Umar, A. M., Linus, O. U., *et al.* Comprehensive review of artificial neural network applications to pattern recognition. *IEEE Access*, **7**, 158820–158846, 2019. [15](#)
- [65] Juan, N. P., Valdecantos, V. N. Review of the application of artificial neural networks in ocean engineering. *Ocean Engineering*, **259**, 111947, 2022. [15](#)

- [66] Emmert-Streib, F., Yang, Z., Feng, H., Tripathi, S., Dehmer, M. An introductory review of deep learning for prediction models with big data. *Frontiers in Artificial Intelligence*, **3**, 4, 2020. [15](#), [20](#), [21](#), [24](#)
- [67] What is computer vision?, 2022. URL <https://www.ibm.com/topics/computer-vision>. [20](#)
- [68] Cloud, G. Descripción general del ajuste de hiperparámetro, 2023. URL <https://cloud.google.com/ai-platform/training/docs/hyperparameter-tuning-overview?hl=es-419>. [21](#)
- [69] Coding, K. ¿qué es la regularización en red convolucional?, 2022. URL <https://keepcoding.io/blog/regularizacion-red-convolucional/>. [25](#)
- [70] He, K., Zhang, X., Ren, S., Sun, J. Deep residual learning for image recognition. En: Proceedings of the IEEE conference on computer vision and pattern recognition, págs. 770–778. 2016. [29](#)
- [71] Simonyan, K., Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. [29](#)
- [72] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L. Imagenet: A large-scale hierarchical image database. En: 2009 IEEE conference on computer vision and pattern recognition, págs. 248–255. Ieee, 2009. [29](#)
- [73] Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M. A closer look at spatiotemporal convolutions for action recognition. En: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, págs. 6450–6459. 2018. [30](#)
- [74] Ji, S., Xu, W., Yang, M., Yu, K. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, **35** (1), 221–231, 2012.
- [75] Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M. Learning spatio-temporal features with 3d convolutional networks. En: Proceedings of the IEEE international conference on computer vision, págs. 4489–4497. 2015. [30](#)
- [76] Raiz del error medio cuadrático, 2022. URL <https://www.statisticshowto.com/probability-and-statistics/regression-analysis/rmse-root-mean-square-error/>. [31](#)
- [77] Raiz del error medio cuadrático, 2022. URL <https://www.ibm.com/docs/es/cloud-paks/cp-data/3.5.0?topic=overview-mean-squared-error>. [31](#)

- [78] Error medio absoluto, 2022. URL <https://www.ibm.com/docs/es/cloud-paks/cp-data/3.5.0?topic=overview-mean-absolute-error>. 31
- [79] Coeficiente de correlación de pearson, 2022. URL <https://www.probabilidadyestadistica.net/coeficiente-de-correlacion-de-pearson/>. 32
- [80] Coeficiente de correlación de pearson, 2022. URL <https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php>. 32
- [81] R2, 2023. URL <https://www.ibm.com/docs/en/cognos-analytics/11.1.0?topic=terms-r2>. 32
- [82] Turney, S. Coefficient of determination (r^2) — calculation interpretation, 2022. URL <https://www.scribbr.com/statistics/efficient-of-determination/>. 32
- [83] Scarlett, R. Why python keeps growing, explained, 2022. URL <https://github.blog/2023-03-02-why-python-keeps-growing-explained/#:~:text=It%20has%20high%20corporate%20demand,in%202022%20by%20recruiters%20worldwide>. 33
- [84] Python: qué es y por qué deberías aprender a utilizarlo, 2021. URL <https://www.becas-santander.com/es/blog/python-que-es.html>. 33
- [85] Bobbia, S., Macwan, R., Benerezeth, Y., Mansouri, A., Dubois, J. Unsupervised skin tissue segmentation for remote photoplethysmography, 2017. URL <https://sites.google.com/view/ybenerezeth/ubfcrppg>. 35, 43
- [86] Descripción general del producto de cloud storage, 2023. URL <https://cloud.google.com/storage/docs/introduction?hl=es-419#:~:text=Cloud%20Storage%20is%20a%20service,your%20projects%20under%20an%20organization>. 41
- [87] Gillmor, C. How does a digital camera sensor work?, 2018. URL <https://medium.com/tech-update/how-does-a-digital-camera-sensor-work-1342974250fd>. 44
- [88] Y, N., YC, N. Photoplethysmography signal analysis for optimal region-of-interest determination in video imaging on a built-in smartphone under different conditions. *Sensors (Basel)*, 10 2017. 44

- [89] Alafeef, M. Smartphone-based photoplethysmographic imaging for heart rate monitoring. *Journal of Medical Engineering & Technology*, **41** (5), 387–395, 2017. URL <https://doi.org/10.1080/03091902.2017.1299233>, pMID: 28300460. [47](#), [48](#)
- [90] Viet, H., ParkJin-Hyeok, LeeSuk-Hwan, KwonKi-Ryong. Real-time heart rate measurement based on photoplethysmography using android smartphone camera. *Journal of Korea Multimedia Society*, **20** (2), 234–243, 02 2017. [47](#)
- [91] Chatterjee, A., Roy, U. K. Ppg based heart rate algorithm improvement with butterworth iir filter and savitzky-golay fir filter. En: 2018 2nd International Conference on Electronics, Materials Engineering Nano-Technology (IEMENTech), págs. 1–6. 2018. [47](#), [48](#), [52](#)
- [92] Chung, H., Ko, H., Lee, H., Lee, J. Deep learning for heart rate estimation from reflectance photoplethysmography with acceleration power spectrum and acceleration intensity. *IEEE Access*, **PP**, 1–1, 03 2020.
- [93] Shyam A, P. S. J. J. S. M., Ravichandran V. Ppgnet: Deep network for device independent heart rate estimation from photoplethysmogram. págs. 1899–1902, 2019. [52](#)
- [94] Parra Barrero, E. Aceleración del algoritmo de viola-jones mediante rejillas de procesamiento masivamente paralelo en el plano focal. *Trabajo Fin de Grado en Ingeniería Electrónica Robótica y Mecatrónica* (pp. 68), 2015. [58](#)
- [95] OpenCV. Optical flow y el algoritmo lucas-kanade, 2022. URL https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html#:~:text=Lucas%2DKanade%20method%20computes%20optical,the%20points%20in%20the%20frame. [59](#)
- [96] Mediapipe, 2023. URL <https://google.github.io/mediapipe/>. [59](#)
- [97] Mediapipe face detection, 2023. URL https://google.github.io/mediapipe/solutions/face_detection.html. [59](#)
- [98] Zhao, A. X. A. Applying video magnification techniques to the visualization of blood flow. Tesis Doctoral, Massachusetts Institute of Technology, 2015. [60](#)
- [99] Tensorflow. Escribir callbacks de keras personalizados, 2023. URL https://www.tensorflow.org/guide/keras/custom_callback?hl=es-419. [64](#)
- [100] Van Gent, P., Farah, H., Van Nes, N., Van Arem, B. Heartpy: A novel heart rate algorithm for the analysis of noisy signals. *Transportation research part F: traffic psychology and behaviour*, **66**, 368–378, 2019. [69](#)