



UNIVERSIDAD DE GRANADA

TRABAJO FIN DE GRADO
INGENIERÍA INFORMÁTICA

Identificación forense a partir de fotografías de caras y cráneos por medio de Deep Learning

Autor
Mario Villar Sanz

Directores
Pablo Mesejo Santiago
Andrea Valsecchi



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE
TELECOMUNICACIÓN

—
Granada, a 7 de Septiembre de 2022

Identificación forense a partir de fotografías de caras y cráneos por medio de Deep Learning

Mario Villar Sanz

Palabras clave: *antropología forense, aprendizaje automático, aprendizaje profundo, visión por computador, identificación humana forense, clasificación de imágenes, redes neuronales convolucionales, redes neuronales siamesas.*

Resumen

Este trabajo de fin de grado (TFG) se ocupa de la identificación humana forense (IDH) a partir de fotografías craneales y faciales de una persona, es decir, aborda la identificación craneofacial directa. Esta tarea, la IDH, es una de las principales en el ámbito de la Antropología Forense, de gran importancia social, política y legal. Aunque existen numerosos estudios en el campo de la IDH craneofacial, la mayoría requiere la anotación de puntos de referencia (*landmarks*) en cráneo y cara, y el consiguiente cálculo del solapamiento de ambas estructuras. En este trabajo se pretende abordar la IDH mediante la aplicación directa de modelos de Aprendizaje Profundo (*Deep Learning*, DL) que permitirán simplificar y acelerar el proceso de identificación. Se trata de una aproximación tremadamente novedosa y original, en la que hay una única publicación previa, y también compleja, dada la gran limitación existente en cuanto a datos de entrenamiento disponibles.

Este TFG se ha valido de un conjunto de modelos craneales 3D de 114 individuos diferentes, así como del conjunto de imágenes faciales UTKFace, de acceso público y con más de 20.000 imágenes. En primer lugar, se han explorado pormenorizadamente ambos conjuntos de datos, y se han extraído imágenes 2D craneales a partir de modelos 3D. A continuación, se han propuesto cinco modelos diferentes de DL, basados en Redes Neuronales Convolucionales y Redes Neuronales Siamesas, y se ha realizado un amplio estudio experimental para compararlos. De los resultados obtenidos se puede concluir que las técnicas de DL empleadas consiguen reducir en un 88 % la muestra de candidatos en la identificación, permitiendo mantener siempre, dentro del 12 % restante la identidad positiva buscada. Como consecuencia, este trabajo ofrece un futuro prometedor para la aplicación de métodos de DL en la reducción de candidatos de la IDH craneofacial directa.

Forensic Identification based on face and skull photographies via Deep Learning

Mario Villar Sanz

Keywords: *forensic anthropology, machine learning, deep learning, computer vision, forensic human identification, image classification, convolutional neural networks, siamese neural networks*

Abstract

This project copes with forensic human identification (IDH) based on skull and face photographies, this is, it approaches direct craniofacial identification. This task, IDH, is one of the main jobs in the scope of Forensic Anthropology, having a big social, political and legal impact. Although numerous studies have been carried out in the field of craniofacial IDH, most of them require landmark annotation in the skull and the face images, and the consequent computation of their overlay. This work aims to address IDH via direct application of Deep Learning (DL) models, which will simplify and speed up the identification process. It is a tremendously novel and original approximation, for which only one previous research has been published, and also a complex one, due to the considerable limitations on available training data.

This TFG has made use of a dataset containing 3D cranial models and 2D face images belonging to 114 different individuals, as well as of the publicly-available 2D face image dataset UTKFace, with over 20,000 images. Firstly, both datasets were explored in detail and 2D skull images were extracted from the 3D cranial models. Following that, five different DL models, based on Convolutional Neural Networks and Siamese Neural Networks, were proposed and a wide experimental analysis was carried out to compare their performance. From the obtained results, it can be concluded that the employed DL techniques are able to reduce by an 88 % the candidate sample while keeping the positive identification in the remaining 12 % of the sample. As consequence, this work shows an optimistic and promising future for the application of DL methods in candidate reduction within direct craniofacial IDH.

Yo, **Mario Villar Sanz**, alumno de la titulación Ingeniería Informática de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 71744694N, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Mario Villar Sanz

Granada a 7 de Septiembre de 2022.

D. **Pablo Mesejo Santiago**, Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

D. **Andrea Valsecchi**, Investigador Senior en Panacea Cooperative Research y miembro del Instituto Andaluz Interuniversitario en Ciencia de Datos e Inteligencia Computacional.

Informan:

Que el presente trabajo, titulado *Identificación forense a partir de fotografías de caras y cráneos por medio de Deep Learning*, ha sido realizado bajo su supervisión por **Mario Villar Sanz**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 7 de Septiembre de 2022.

Los directores:

Pablo Mesejo Santiago

Andrea Valsecchi

Agradecimientos

A mis tutores, Pablo y Andrea, por su esfuerzo y guía a lo largo de todo este proyecto. Tanto por su paciencia como por la confianza que han puesto en mí.

A mis padres, por su apoyo y ayuda a lo largo de este camino y, en especial, por su trabajo para mejorar mi futuro. También a mi hermana, por su ánimos en los mejores y peores momentos.

Índice general

| | |
|--|-------------|
| Resumen | I |
| Abstract | III |
| Índice de figuras | XIII |
| Índice de tablas | XVII |
| 1. Introducción | 1 |
| 1.1. Descripción del problema | 1 |
| 1.2. Motivación | 5 |
| 1.3. Objetivos | 7 |
| 2. Fundamentos Teóricos | 9 |
| 2.1. Ciencias Forenses | 9 |
| 2.1.1. Identificación de restos humanos | 10 |
| 2.1.2. Antropología Forense | 11 |
| 2.1.3. Ciencias Forenses Digitales | 11 |
| 2.2. Aprendizaje Automático | 12 |
| 2.2.1. Tipos de aprendizaje | 13 |
| 2.2.2. Redes Neuronales Artificiales | 14 |
| 2.3. Deep Learning | 17 |
| 2.3.1. Redes Neuronales Convolucionales | 18 |
| 2.3.2. Redes Neuronales Siamesas | 21 |
| 2.3.3. Otros conceptos relevantes | 25 |
| 2.4. Visión por Computador | 27 |
| 3. Estado del arte | 29 |
| 3.1. La clasificación de imágenes con métodos de Deep Learning | 29 |
| 3.2. Reconocimiento facial mediante Deep Learning | 30 |
| 3.3. Identificación humana directa basada en imágenes digitales faciales y craneales | 33 |
| 4. Planificación e Implementación | 35 |
| 4.1. Planificación temporal y económica | 35 |
| 4.2. Diseño e implementación | 37 |
| 4.3. Entorno de ejecución | 41 |
| 5. Materiales y Métodos | 45 |
| 5.1. Conjunto de datos | 45 |
| 5.1.1. Conjunto de datos craneales | 45 |
| 5.1.2. Conjunto de datos UTKFace | 48 |
| 5.2. Métodos | 48 |
| 5.2.1. Procesamiento del conjunto de datos craneales | 49 |

| | |
|--|------------|
| 5.2.2. FaceNet | 55 |
| 5.2.3. Modelos implementados para realizar la identificación craneofacial a partir de fotografías de cráneos y caras | 56 |
| 5.2.4. Modelos propuestos | 71 |
| 6. Experimentación | 75 |
| 6.1. Protocolo de validación experimental | 75 |
| 6.2. Separación en entrenamiento, validación y test | 77 |
| 6.3. Test de los modelos y métricas de error | 77 |
| 6.4. Generación de conjuntos de imágenes 2D | 81 |
| 6.4.1. Generación del dataset de imágenes 2D craneales . . . | 81 |
| 6.4.2. Data augmentation de imágenes faciales | 84 |
| 6.4.3. Generación de la Base de Datos de imágenes faciales de Test | 87 |
| 6.5. Experimentación con los modelos propuestos | 88 |
| 6.5.1. Elección del modelo base y optimizador | 88 |
| 6.5.2. Selección de hiperparámetros en hold-out | 90 |
| 6.5.3. Evaluación de modelos en Validación Cruzada y discusión de resultados | 93 |
| 7. Conclusiones y trabajos futuros | 103 |
| Bibliografía | 107 |
| Apéndice A | 119 |
| Apéndice B | 121 |
| Apéndice C | 123 |

Índice de figuras

| | | |
|-------|--|----|
| 1.1. | La AF se encarga del estudio de huesos humanos para su aplicación en el ámbito médico-legal | 2 |
| 1.2. | La relación entre el cráneo y la cara humana | 3 |
| 1.3. | Proceso de identificación basado en Superposiciones Craneofaciales | 4 |
| 1.4. | El problema de la identificación humana forense en cifras . . | 5 |
| 1.5. | Muestreo del cráneo por radiografía, CT y fotografía | 6 |
| 2.1. | Estructura ósea del cráneo humano | 12 |
| 2.2. | Esquema de la superposición frontal de la cara sobre el cráneo humano | 12 |
| 2.3. | Esquema de la superposición lateral de la cara sobre el cráneo humano | 13 |
| 2.4. | Estructura de una Red Neuronal Artificial básica | 15 |
| 2.5. | Comparativa de las capas totalmente conectadas y las capas convolucionales en Redes Neuronales | 19 |
| 2.6. | Convolución de un filtro sobre una imagen | 20 |
| 2.7. | Funcionamiento de una capa convolucional | 20 |
| 2.8. | Varios tipos de capas de Pooling | 22 |
| 2.9. | Capa de Dropout | 22 |
| 2.10. | Arquitectura de una Red Neuronal Siamesa con tres sub redes neuronales | 23 |
| 2.11. | Arquitectura de una Red Neuronal Siamesa con dos sub redes neuronales y salida sigmoidal | 25 |
| 2.12. | Ejemplos de la generación de nuevas imágenes faciales a partir de una original | 27 |
| 2.13. | Bounding box de una esfera | 28 |
| 3.1. | Número de artículos publicados por año en el ámbito del DL y las imágenes desde 2000 | 30 |
| 3.2. | Evolución de las publicaciones en los últimos años para algunos de los diseños de redes convolucionales más utilizadas . . | 32 |
| 4.1. | Diagrama de Gantt con la planificación inicial del TFG. . . . | 36 |
| 4.2. | Diagrama de Gantt con la planificación real del TFG. . . . | 36 |
| 4.3. | Diseño e implementación del presente TFG. | 38 |
| 4.4. | Metodología de desarrollo en cascada empleada en este proyecto. . | 39 |
| 4.5. | Interdependencias presentadas entre los diferentes archivos que componen el sistema diseñado e implementado. | 42 |
| 4.6. | Diagrama de secuencia del sistema implementado, que muestra el proceso de interacción entre los diferentes archivos. . . | 42 |
| 5.1. | Diferentes ejemplos de los modelos craneales del conjunto de datos disponible | 47 |

| | | |
|-------|---|----|
| 5.2. | Ejemplos de las imágenes faciales ya recortadas y alineadas del <i>dataset</i> UTKFace. | 48 |
| 5.3. | Selección de tres puntos en cada malla para extraer la transformación que alinee el modelo de la izquierda con el modelo de la derecha | 49 |
| 5.4. | Ejemplo de las imperfecciones presentes en algunos modelos 3D craneales del conjunto de datos | 50 |
| 5.5. | El centro de un cráneo es el punto medio de los extremos del modelo 3D en cada eje | 51 |
| 5.6. | Diferentes perspectivas de un cráneo en función de distancia de la cámara al mismo | 52 |
| 5.7. | Ilustración de la obtención de los ángulos de visión mínimos (horizontal y vertical) | 53 |
| 5.8. | Ilustración de la obtención del máximo desplazamiento positivo aplicable en el eje X | 56 |
| 5.9. | Estructura de los módulos Inception | 57 |
| 5.10. | Proceso de predicción de las imágenes craneales en las Redes Neuronales Siamesas entrenadas con Triplet Loss | 59 |
| 5.11. | Hay tres escenarios posibles en las tripletas de entrenamiento: <i>negativos sencillos</i> , <i>negativos semi-hard</i> y <i>negativos hard</i> | 60 |
| 5.12. | La selección online de tripletas consiste en eliminar aquellas tripletas sencillas y solo computar la función de pérdida sobre las tripletas semi-hard y hard | 62 |
| 5.13. | La estrategia generación online de tripletas batch-all consiste en emparejar cada imagen facial con todas las imágenes craneales positivas del mismo individuo, y todos estos pares, a su vez, con todas las imágenes craneales negativas del batch . | 64 |
| 5.14. | La estrategia generación online de tripletas batch-hard consiste en emparejar cada imagen facial con la imagen craneal positiva y la negativa más complicadas | 65 |
| 5.15. | Arquitectura de Red Neuronal Siamesa propuesta | 66 |
| 5.16. | Bloque de capas propuesto y utilizado en el modelo de salida | 66 |
| 5.17. | Proceso de predicción de las imágenes craneales en las Redes Neuronales Siamesas con salida sigmoidal | 67 |
| 5.18. | Histogramas de la distribución de distancias entre imágenes de pares positivos y entre imágenes de pares negativos | 69 |
| 5.19. | El umbral de decisión divide ambos histogramas de distancias (entre pares positivos y entre pares negativos) en dos | 70 |
| 5.20. | Ejemplo de árbol de decisión en para la elección del umbral de decisión en el escenario de una Red Neuronal Siamesa entrenada con Triplet Loss | 71 |

| | | |
|-------|---|-----|
| 6.1. | Protocolo de CV con 5 folds: en cada iteración uno de los folds es utilizado como test y los 4 restantes como entrenamiento y validación | 76 |
| 6.2. | Protocolo de hold-out: el conjunto de datos es únicamente dividido en entrenamiento, validación y test una única vez | 76 |
| 6.3. | La curva CMC permite visualizar en qué medida se ha conseguido reducir la muestra de posibles candidatos | 79 |
| 6.4. | La curva NFA permite visualizar el grado de confianza con el que se rechaza una imagen negativa | 81 |
| 6.5. | Perspectiva frontal del modelo 3D del cráneo del individuo PM11, modelo utilizado para el alineamiento de todos las mallas del conjunto de datos. | 82 |
| 6.6. | Modelo 3D correspondiente al individuo PM1, ligeramente no alineado | 82 |
| 6.7. | Modelo 3D correspondiente al individuo PM32, notoriamente no alineado | 83 |
| 6.8. | Modelo 3D correspondiente al individuo PM73, que presenta una imperfección que ha sido eliminada durante el preprocesamiento | 83 |
| 6.9. | Las imágenes 2D son generadas a partir de los modelos 3D craneales tras aplicar rotaciones y desplazamientos aleatorios sobre el modelo y distanciando la cámara de él | 84 |
| 6.10. | Varios ejemplos de tripletas de imágenes | 85 |
| 6.11. | Ejemplos de pares de imágenes faciales-craneales | 86 |
| 6.12. | Ejemplos del <i>data augmentation</i> realizado sobre las imágenes faciales de los individuos positivos. | 86 |
| 6.13. | Dada la naturaleza de los experimentos de CV, es necesario disponer de una BD facial diferente para cada fold | 87 |
| 6.14. | La BD facial de test en hold-out está compuesta por las imágenes faciales positivas de test (IM_{Test}), las imágenes faciales de validación (IM_{Val}) y completada hasta las 100 imágenes con parte del <i>dataset</i> UTKFace. | 88 |
| 6.15. | Flujo de trabajo seguido en el entrenamiento y test de las Redes Neuronales Siamesas | 89 |
| 6.16. | Curvas CMC de los modelos con mayor reducción de la muestra de candidatos (menor Acc_k^1) | 95 |
| 6.17. | Curvas NFA de los modelos con mayor reducción de la muestra de candidatos (menor Acc_k^1) | 96 |
| 6.18. | Curva CMC (6.18a) y NFA (6.18b) del modelo que consigue mayor accuracy | 98 |
| 6.19. | Evolución de los umbrales de decisión obtenidos con los tres métodos propuestos a medida que el modelo de DL aprende a separar las distancias entre pares positivos y pares negativos | 101 |

| | |
|---|-----|
| 6.20. Ejemplo de modelo de Red Neuronal Siamesa entrenada con Triplet Loss, colapsado durante el entrenamiento | 102 |
| 7.1. Estructura de Red Neuronal Siamesa entrenada con Contrastive Loss propuesta como trabajo futuro | 104 |
| 7.2. Estructura de Red Neuronal Convolucional que recibe dos imágenes concatenadas en un único volumen, propuesta como trabajo futuro | 105 |

Índice de tablas

| | |
|--|-----|
| 3.1. Principales diseños de redes neuronales profundas utilizadas en la clasificación de imágenes. | 31 |
| 5.1. Estadísticos de los modelos 3D craneales del conjunto de datos utilizado | 46 |
| 5.2. Arquitectura de FaceNet | 57 |
| 6.1. Parrilla de valores de cada hiperparámetro para cada uno de las arquitecturas analizadas | 91 |
| 6.2. Resultados de la batería experimental de hold-out para la preselección de hiperparámetros | 92 |
| 6.3. Resultados de los experimentos finales en CV, mostrando el mejor modelo de cada arquitectura según el tamaño de la muestra reducida | 94 |
| 6.4. Resultados de los experimentos finales en CV, mostrando el mejor modelo de cada arquitectura según el accuracy total . . | 97 |
| 6.5. Resultados de los experimentos finales en CV, mostrando el mejor modelo de cada arquitectura según el accuracy en test positivo | 99 |
| 6.6. Resultados de los experimentos finales en CV, mostrando el mejor modelo de cada arquitectura según el accuracy en test negativo | 99 |
| 7.1. Información extendida relacionada con el conjunto de modelos 3D craneales utilizado | 119 |

Capítulo 1

Introducción

1.1. Descripción del problema

Las Ciencias Forenses (CF) aúnan el conocimiento y las metodologías de diversas disciplinas para su aplicación en la resolución de casos jurídicos [49, 120]. Su objetivo es establecer evidencias¹ fundamentándose en el análisis de las personas, sitios, situaciones y objetos en el ámbito médico-legal. En otras palabras, las CF son el conjunto de disciplinas cuyo objeto común es el de la materialización de pruebas a efectos judiciales mediante una metodología científica. En este sentido, cualquier ciencia se convierte en forense en el momento que sirve al procedimiento judicial.

Uno de los campos más importantes hoy en día de las CF es la Antropología Forense (AF) [23]. La AF consiste en el estudio del esqueleto humano con objetivos médico-legales [121]. Los expertos en esta materia analizan los huesos humanos con el objetivo de hallar toda la información posible tanto de los individuos con los que se corresponden como de las circunstancias relacionadas con su situación o su muerte, tal como se esquematiza en la Figura 1.1. Entre las actividades de la AF se incluyen, por ejemplo, la estimación del perfil biológico de una persona, la determinación de las causas de lesiones óseas o la recuperación de restos óseos manteniendo el mayor grado de información sobre estos.

En concreto, un campo de especial interés debido a su gran impacto social y policial es la identificación humana (IDH): establecer la identidad de una persona. Las metodologías actuales más utilizadas continúan siendo la comparación de huellas dactilares y las pruebas de ADN. La fiabilidad de estas pruebas es elevada, pero presentan algunos inconvenientes: tienen un

¹Una evidencia es una prueba que permite demostrar la verdad sobre un hecho en el ámbito legal.

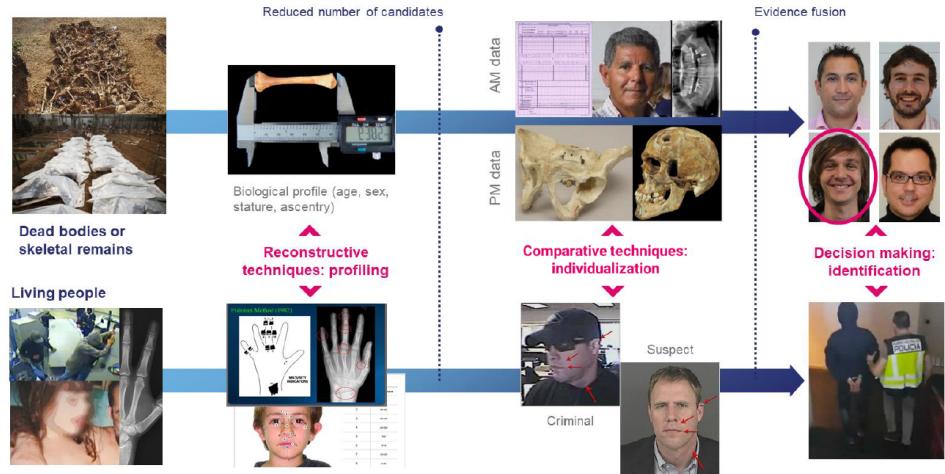


Figura 1.1: La AF se encarga del estudio de los huesos humanos con el objetivo de extraer información sobre las personas en el ámbito médico-legal. Dicho estudio puede servir de ayuda tanto en la identificación de individuos muertos (fila superior) como de individuos vivos (fila inferior). Imagen extraída de [81].

coste económico alto², requieren de tiempo para llevar a cabo el proceso³ y, principalmente, exigen registros previos del individuo, es decir, una segunda muestra, para poder realizar la identificación (registros de huellas dactilares o ADN del individuo) que no siempre están disponibles [119].

Las técnicas de AF ofrecen una alternativa resiliente al estar basadas en el estudio de tejidos duros [27]. Estos tejidos sobreviven, en general, a los procesos de descomposición que sufre el cadáver humano (tanto por descomposición natural del cadáver como por descomposición no natural, debido a sucesos como incendios), pudiendo ser objeto de estudio durante espacios temporales mayores. Entre los elementos identificativos del esqueleto humano, el cráneo constituye uno de los principales, por lo que supone una de las piezas clave en la identificación, especialmente de cadáveres. Por esta razón, la IDH a partir de comparativas cráneo - cara (ver Figura 1.2) es una línea de estudio importante en la AF [26]. Recientemente, hay estudios que avalan la posible aplicación de técnicas de Inteligencia Artificial (IA) a esta tarea [81].

En la última década, las técnicas de IA han permitido la resolución de

²El coste de la identificación por ADN a partir de huesos es, aproximadamente, 500€ por comparación. En el caso de una máquina AFIS, empleada para la identificación por huella dactilar, el coste de adquirir una de ellas es del orden de cientos de miles o de millones de euros.

³En el caso de múltiples comparaciones el proceso puede llegar a demorarse incluso varios días.

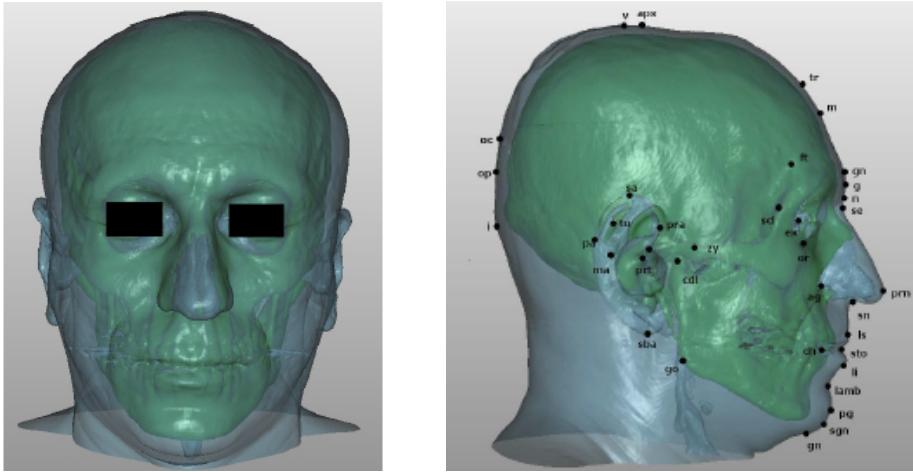


Figura 1.2: El cráneo y la cara humana están estrechamente asociados, encan-jando entre sí de forma precisa y constituyendo un elemento de identificación relevante. Imágenes extraídas de [26].

complicadas tareas gracias a los avances, entre otros, en el Aprendizaje Automático (AA) y el Aprendizaje Profundo (*Deep Learning*, DL) [38, 68, 103]. Aplicaciones como reconocimiento automático de dígitos manuscritos [37] o software de IA capaz de ganar a campeones humanos en el Go [109] remarcan la importancia de los avances conseguidos. Particularmente, gracias a las mejoras en el hardware de los ordenadores (incremento de la memoria de almacenamiento, mejora de la capacidad de procesamiento de las CPUs y las GPUs, etc.), la Visión por Computador (VC) se ha visto notablemente impulsada por la aplicación de los métodos de DL. Todas estas disciplinas han abierto un gran abanico de posibilidades de resolución de problemas dentro del ámbito médico y forense [9, 34], entre los que se incluye la IDH.

La aplicación de métodos de AF en conjunción con técnicas de VC ha permitido conseguir notables avances en IDH a partir de imágenes y mo-delos (tanto 2D como 3D) del cráneo de una persona [18]. El método más utilizado en la actualidad, dentro de lo que es IDH craneofacial, es la Superposición Craneofacial (*Craniofacial Superimposition*, CFS). Este proce-dimiento tienen varias desventajas intrínsecas: por un lado, está supeditado a la estimación del grosor de los tejidos blandos de la cabeza y, por otro lado, genera un resultado que debe ser interpretado posteriormente (un so-lapamiento cara-cráneo que de ser analizado para decidir si las imágenes originales pertenecen a la misma persona). Estas tareas son, a día de hoy, tareas complejas, subjetivas, lentas, dependientes de la experiencia de un experto y difícilmente reproducibles [53].

Actualmente existen múltiples trabajos en este campo [52, 122, 124, 133] que utilizan modelos del cráneo y de la cara para automatizar parte del pro-

ceso de CFS y realizar identificaciones. Estos estudios se centran en la automatización del cálculo del solapamiento cráneo-cara (*Skull-Face Overlay*, SFO), estudiando posteriormente la calidad del solapamiento obtenido para decidir si se corresponden con la misma persona o no (ver Figura 1.3). No obstante, la adquisición y el procesado de los materiales necesarios es una tarea compleja y la propia CFS es poco eficiente en tiempo y en coste, ya que exige una etapa no automatizada para la interpretación del solapamiento. En este TFG se propone un enfoque totalmente rupturista, disruptivo y novedoso basado en la comparación directa de imágenes 2D-2D a través de una red neuronal, sustituyendo así la CFS del anterior método. Además, supone una gran mejora en la adquisición y preprocesado⁴ de los materiales, fotografías 2D craneales y faciales, ya que esta etapa se agiliza: no es necesario, por ejemplo, añadir landmarks faciales y craneales, entre otros.

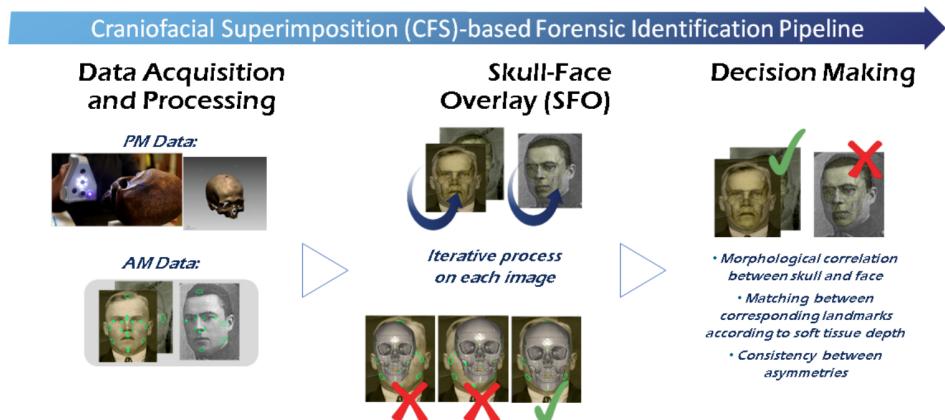


Figura 1.3: Proceso de identificación basado en CFS. Imagen extraída de [81].

En este trabajo de fin de grado se pretende explorar la identificación humana a partir de imágenes craneales e imágenes faciales empleando técnicas de DL y VC. Este proyecto propone relacionar una imagen 2D postmortem⁵ (PM) de la vista frontal craneal de un cadáver humano con una fotografía facial antemortem⁶ (AM) correspondiente al mismo individuo. Una novedad presentada en este trabajo, respecto a otro trabajos previos en el mismo campo [14, 18, 26, 52, 124], es la supresión de la etapa del cálculo del solapamiento cara-cráneo por medio de registrado de imágenes, lo cual supone una ventaja a nivel computacional.

⁴El preprocesado de los materiales se refiere a aquellas etapas previas a la realización del método en las que se adaptan los datos brutos disponibles para que su utilización posterior sea más sencilla. El objetivo final de estas etapas es aumentar la calidad de los datos para mejorar el rendimiento del método que los utiliza.

⁵Postmortem se refiere a lo ocurrido después de la muerte de la persona.

⁶Antemortem se refiere a lo ocurrido antes de la muerte de la persona.

1.2. Motivación

La desaparición de personas es un problema importante a nivel mundial, con un gran número de desaparecidos a lo largo de las últimas décadas [89]. Solo en España la cifra de desaparecidos se tasa en más de 200.000 [42]. La IDH es, por tanto, una tarea de gran impacto en la sociedad.

La Figura 1.4 ilustra el problema de la desaparición humana en el mundo. Entre las principales causas de desapariciones se encuentran los conflictos bélicos y los desastres naturales, cuyos efectos afectan en mayor medida a zonas con bajo desarrollo económico y tecnológico [54, 86, 100, 123]. La complejidad de la IDH, difícil de por sí, se extrema en estas situaciones: el escaso progreso tecnológico de estas regiones se une a la destrucción de infraestructuras (dificultando, por ejemplo, el trabajo en laboratorios adecuadamente equipados), al gran incremento en un escaso espacio de tiempo del número de personas sin identificar, y a la escasez de información sobre las personas desaparecidas (por la descomposición y desaparición de los restos humanos, la baja disponibilidad de registros previos con los que comparar y realizar identificaciones, etc.). Por todo lo anterior, resulta de interés disponer de métodos de IDH cuya eficiencia en tiempo y en coste económico sea baja, que presenten una alta resiliencia frente a la descomposición del cadáver humano, que sean lo más robustos, reproducibles y objetivables posibles, y donde la toma de muestras sea un proceso sencillo y accesible.

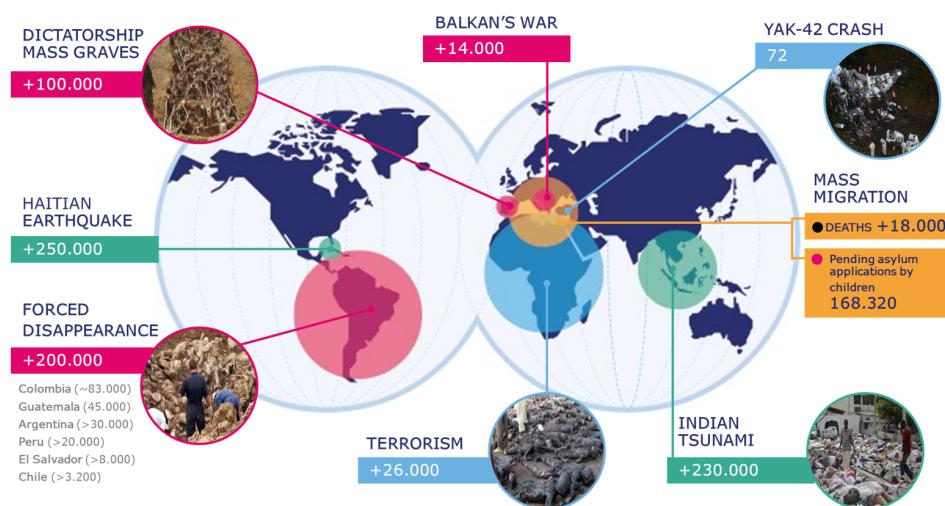


Figura 1.4: El problema de la IDH en cifras. Imagen elaborada por Panacea Cooperative Research e incluida con permiso de los autores.

Las técnicas de AF ofrecen la posibilidad de realizar la IDH a partir de materiales que perduran en el tiempo. El cráneo es uno de estos elementos y es utilizado en la IDH a través de reconstrucciones faciales 2D, recons-

trucciones faciales 3D y CFS sobre imágenes 2D [119]. Estos procedimientos son tradicionalmente realizados por un experto en AF y toman varios días en realizarse. No obstante, los avances en la automatización del cálculo del solapamiento cara-cráneo de la CFS han conseguido reducir los tiempos de la etapa de SFO muy notablemente, pasando de media hora o más a unos milisegundos, mientras se sigue aumentando la certidumbre y robustez de los resultados [124]. La automatización de la CFS para la IDH cumple con todas las condiciones especificadas más arriba:

- Su accesibilidad es elevada. Gracias a la expansión de los dispositivos móviles y las cámaras, la toma de imágenes es muy sencilla, facilitando la adquisición de los materiales necesarios para realizar las identificaciones. Este aspecto es especialmente importante en los escenarios bélicos y de desastres naturales descritos anteriormente, donde la adquisición de otros materiales más complejos, como radiografías y tomografías computarizadas (Computed Tomography, CT) (ver Figura 1.5), conlleva mayor dificultad.
- Su eficiencia en tiempo es alta y su coste económico es bajo.
- Su resiliencia es elevada (la descomposición del cráneo humano es muy lenta). Además, es fácilmente reproducible, dada su gran accesibilidad, y objetivable, ya que no depende de aspectos subjetivos (como podría ser la comparación de una reconstrucción facial con otras imágenes faciales por un experto forense).



Figura 1.5: Tres métodos de muestreo de un cráneo, de izquierda a derecha: por radiografía, por modelado 3D a partir de CT y por fotografía. La CT es el método más costoso, seguido de la radiografía y siendo la fotografía el más sencillo de adquirir de todos. Imágenes extraídas de [77], elaboración propia y de [51].

Sin embargo, la automatización de la CFS sigue presentando, actualmente, la necesidad de la intervención de un experto en alguna etapa del proceso

de identificación, por ejemplo, en el marcado de landmarks craneales y faciales. Este TFG propone un **método de IDH directa fundamentado en Redes Neuronales que comparan fotografías faciales y fotografías craneales**. Con ello se pretende suprimir las etapas intermedias presentes en el proceso automatizado de CFS (como el marcado de landmarks mencionado anteriormente), realizando una identificación directa a partir de las imágenes 2D. Este proyecto no solo comparte las ventajas anteriores del automatizado de la CFS, sino que también permitirá **reducir la complejidad** del método al evitar el proceso de registrado evolutivo-difuso del estado del arte para la etapa de SFO [124]. Además, favorecerá una **reducción del coste en tiempo**, dado que se elimina la múltiple proyección del modelo craneal sobre el modelo facial, el costoso marcado de *landmarks* faciales y craneales, la estimación del tejido blando que separa cráneo y cara, y se automatiza la toma de decisión sobre la correspondencia entre imágenes. Finalmente, la supresión de la etapa de marcado de landmarks propiciará un **aumento de la accesibilidad** del proceso de identificación: requiere únicamente el acceso a un computador, sin la necesidad de la supervisión de un experto.

Por todo lo anterior, una solución al problema de la IDH que permita la identificación directa mediante fotografías faciales y craneales podría tener un gran impacto positivo en los ámbitos social y económico.

1.3. Objetivos

El objetivo principal de este trabajo es el siguiente:

Proponer un método de DL para automatizar la identificación forense a partir de fotografías craneales y faciales.

Durante el desarrollo de este proyecto para lograr el anterior objetivo, se pretenden abordar los siguientes objetivos parciales:

- I. Realizar un estudio exhaustivo del estado del arte en identificación humana mediante imágenes craneales y faciales, así como de las tareas relacionadas con ello, para tener una visión completa del desarrollo tecnológico en este campo y poder analizar las mejores herramientas para abordar exitosamente el problema en cuestión.
- II. Diseñar e implementar una herramienta para la exploración y preprocesado de los conjuntos de datos disponibles, compuestos por modelos 3D de cráneos y fotografías faciales frontales. Obtener las imágenes 2D craneales a utilizar en el entrenamiento de los modelos.

- III. Examinar las técnicas de DL más adecuadas al problema y proponer un modelo profundo para automatizar la solución del problema en cuestión.
- IV. Diseño, implementación, entrenamiento y validación experimental del modelo de DL escogido para la identificación de personas a partir de imágenes craneales y faciales, y validar la viabilidad de estas técnicas en la resolución del problema de la IDH.

Capítulo 2

Fundamentos Teóricos

2.1. Ciencias Forenses

Las CF son aquellas en las que se desarrollan técnicas o métodos científicos para buscar evidencias que puedan ser utilizadas en el ámbito legal y de la justicia [49, 94]. Nótese que por ámbito legal y de la justicia no solo se refiere al campo criminal sino también al campo civil y policial.

Un aspecto importante de las CF es la comparación de evidencias con el fin de extraer pruebas concluyentes sobre el caso. Por ejemplo, las técnicas de test de ADN se encargan de comparar ADN encontrado en la escena con registros de ADN con el fin de encontrar sujetos sospechosos, posibles víctimas, etc.: siguiendo un método científico se buscan pruebas que ayuden en el campo policial o de la justicia.

Estándar de Daubert

Un principio legal importante en las CF es el Estándar de Daubert, desarrollado en la década de 1990 y establece las normas y directrices para aceptar en caso legales evidencias generadas mediante métodos científicos [84]. Este Estándar se fundamenta en el Estándar de Frye (de existencia previa) y plantea la aceptación de la evidencia si la técnica con la que fue obtenida cumple las siguientes cinco reglas:

1. La técnica científica es aceptada en la comunidad científica.
2. Se debe conocer el ratio de error de la técnica científica.
3. La técnica científica debe ser comprobable.
4. La técnica científica ha de haber sido publicada y debe ser peer-reviewed.
5. Deben existir controles y estándares científicos aplicables sobre la técnica científica empleada.

Cualquier método de las CF desarrollado debe poder cumplir el Estándar de Daubert para poder ser un método utilizable en el ámbito legal.

2.1.1. Identificación de restos humanos

La identificación de restos humanos es el campo de las CF encargado de determinar información sobre restos humanos no identificados resultantes tanto de crímenes como de cualquier otro suceso como catástrofes naturales [119]. La información extraída a partir de los restos incluye desde la analizar si los restos son restos humanos hasta la estimación del perfil biológico de la persona a la que pertenecen los restos y la propia identificación de la persona.

2.1.1.1. Identificación humana

La IDH se puede definir como el proceso de vinculación de restos humanos no identificados con una persona con identificación conocida [23]. Desde el punto de vista legal, la identificación ha de ser realizada en última instancia por una autoridad medicolegal en base a información recopilada sobre el caso, por ejemplo, una identificación realizada mediante procedimientos científicos aplicados a los restos humanos en cuestión.

Existen múltiples métodos científicos para identificar restos humanos y en general se fundamentan en la comparación de la víctima con una base de datos de individuos, cotejando la información disponible y estableciendo la identificación con la persona con mayor similitud. Los métodos más utilizados los análisis de ADN, el cotejo de huellas dactilares y la comparación dental (a través de radiografías o tomografías computarizadas - CTs) [25]. No obstante, estas técnicas tienen claros inconvenientes en su uso [23, 119]:

- Las tres técnicas requieren de registros previos AM complejos: radiografías o CTs dentales, registro de huella dactilar del individuo o muestras de ADN ¹.
- Los análisis de ADN y la comparación dental son métodos costosos económicamente y temporalmente y, junto con el registro de huellas dactilares, no están disponibles en muchos países o regiones con bajo acceso a estos recursos.
- Las tres técnicas sufren también la descomposición del cuerpo humano: la descomposición de tejidos blandos o la momificación del cuerpo dificulta o inhabilita la toma de huellas dactilares y el muestreo de ADN del sujeto, mientras que la pérdida de dientes o deformación de la mandíbula tiene un gran impacto en la comparación dental.

¹Las muestras de ADN también pueden ser comparadas con muestras de familiares de víctimas

Aparte de los anteriores métodos, existen una gran variedad de técnicas de identificación, entre las que se incluyen procedimientos basados en tejidos blandos, en tejidos duros y en diferentes aspectos característicos de las personas (modificaciones estéticas, forma de escritura, etc.).

La IDH se puede dividir en dos niveles: la identificación positiva y la reducción de la muestra de candidatos. El primer nivel consiste en determinar exactamente la persona correspondiente a los restos humanos, mientras que el segundo nivel radica en la limitación del número de posibles candidatos.

2.1.2. Antropología Forense

La Antropología Forense es el campo de estudio aplicado al ámbito medi-colegal que comprende las técnicas empleadas en el análisis y restauración del esqueleto humano [4, 23, 87]. Básicamente, la AF se encarga de tareas relati-vas a la identificación de restos humanos, desde la perspectiva del esqueleto humano: determinación del perfil biológico de las víctimas, identificación de lesiones traumáticas en huesos, delimitación de intervalos postmortem² e IDH.

La última de las anteriores tareas es la afrontada en este proyecto, en particular, a partir del análisis del cráneo y la cara de la víctima. El cráneo humano está formado por 22 huesos visibles (tres más dentro de cada oído - martillo, yunque y estribo), haciendo un total de 28 huesos que se pueden dividir en dos conjuntos: el neurocráneo (huesos que conforman los laterales, la parte superior y la parte trasera) y el esplacnocráneo (o huesos de la cara), como se puede ver en la Figura 2.1. Por otro lado, la cara humana está cubierta en su mayoría por piel; las características principales presentes en la cara son los ojos, la nariz, los labios y dientes, las cejas, el pelo, las orejas y la barba (en algunas personas). En las Figuras 2.2 y 2.3 se puede ver la correspondencia del cráneo con la cara humana.

2.1.3. Ciencias Forenses Digitales

Desde hace ya décadas, con la generalización del uso de recursos compu-tacionales, las CF han evolucionado hacia modelos híbridos (dando lugar a lo que se conoce como Ciencias Forenses Digitales [36, 107]) basados en la cooperación de expertos humanos con ordenadores y materiales digitales. Esto ha permitido la aplicación de técnicas de AA en el campo de las CF, por ejemplo, para el desarrollo de métodos de clasificación de materiales digitales forenses [78].

Más en concreto, la VC y el DL han permitido avances interesantes en las CF como, por ejemplo, sistemas de detección de falsificación de imágenes digitales [92] y la automatización parcial de CFS [18]. En este trabajo se

²El intervalo PM es el intervalo de tiempo estimado en el que ocurrió la muerte de la víctima.

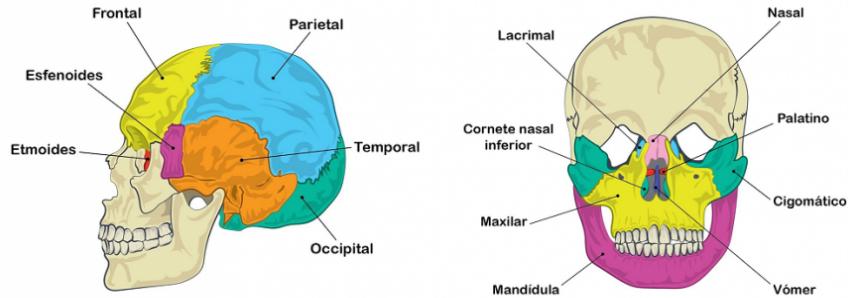


Figura 2.1: Estructura ósea del cráneo humano. En la izquierda se pueden ver los huesos pertenecientes al neurocráneo, en la derecha los pertenecientes al esplacnocráneo. Imagen extraída de [10].

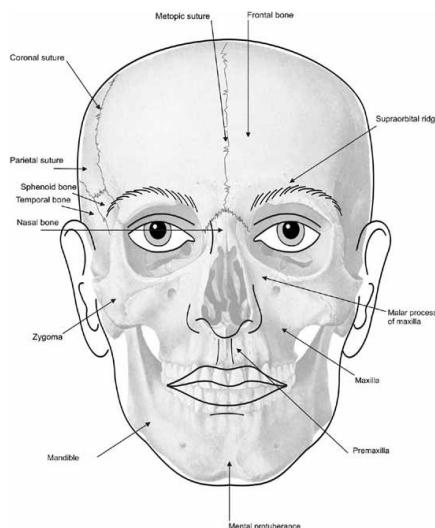


Figura 2.2: Esquema de la superposición frontal de la cara sobre el cráneo humano. Imagen extraída de [7].

espera aplicar técnicas de VC y DL al problema de IDH a partir de imágenes craneales y faciales.

2.2. Aprendizaje Automático

Existen problemas para los cuales no se conoce una solución matemática pero para los que se dispone de datos sobre los que construir una solución empírica [3]. En otras palabras, la solución a estos problemas no puede ser implementada como una secuencia de instrucciones en un ordenador porque

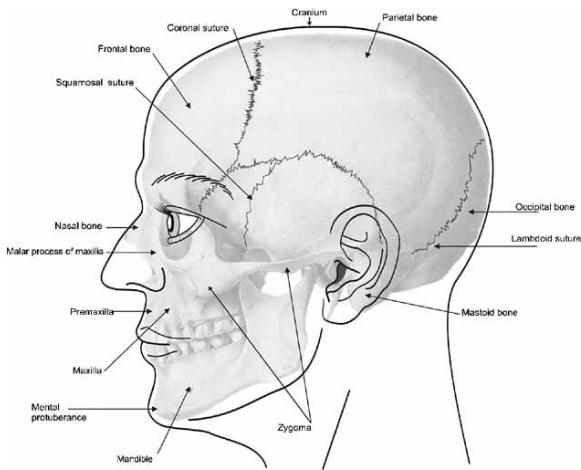


Figura 2.3: Esquema de la superposición lateral de la cara sobre el cráneo humano. Imagen extraída de [7].

el algoritmo de resolución no es conocido, pero existen datos que permiten aprender patrones sobre los problemas en cuestión.

El AA es el área de la IA que busca diseñar soluciones a problemas empleando modelos que aprendan a partir de datos para mejorar su rendimiento en dichos problemas [8, 16]. El modelo que se plantea para la resolución del problema no es más que un modelo matemático con conjunto de parámetros ajustables (hiperparámetros) que son optimizados durante el aprendizaje a partir de los datos.

2.2.1. Tipos de aprendizaje

Dentro de AA se pueden encontrar tres tipos de aprendizaje:

- Aprendizaje Supervisado. Se dispone de un conjunto de datos (*dataset*) con ejemplos explícitos de pares entrada-salida y se desea aprender un modelo que prediga salidas para nuevas entradas.
- Aprendizaje por Refuerzo. Se dispone de un conjunto de datos que solo incluye entradas pero sin contener específicamente las salidas. No obstante, se puede medir el efecto que la aplicación del modelo sobre los datos de entrada de cara a mejorarlo (frecuentemente utilizado para aprender a jugar a un juego, etc.).
- Aprendizaje No Supervisado. Se dispone de un conjunto de datos que solo incluye entrada y no hay ningún tipo de información sobre las salidas del sistema (ni siquiera una forma de medir el desempeño del modelo). El objetivo es extraer información de los datos de cara a detectar características en ellos, agruparlos según su parecido, etc.

Un problema de AA supervisado consiste en determinar la función matemática g que mejor modele el dominio $f : \mathcal{X} \rightarrow \mathcal{Y}$, donde \mathcal{X} es el espacio de las variables de entrada e \mathcal{Y} es el espacio de las variables de salida; $\{\mathcal{X}, \mathcal{Y}\}$ definen el dominio del problema y la función f es desconocida. Normalmente, se dispone de una representación de los valores del dominio mediante un conjunto de datos \mathcal{D} conteniendo N pares de tuplas de valores $(x_1, y_1) \dots (x_N, y_N) \mid y_n = f(x_n) \forall n \in \{1, N\}$. Por medio de un algoritmo de aprendizaje supervisado se busca un modelo $g : \mathcal{X} \rightarrow \mathcal{Y}$ con la mejor aproximación a f dentro de un conjunto \mathcal{H} de posibles modelos. En la práctica las entradas y salidas son vectores de números y la función g es una combinación (lineal o no lineal) de las entradas.

Dentro de los problemas de aprendizaje supervisado es frecuente realizar una división en problemas de regresión y problemas de clasificación. Mientras los problemas de **regresión** están caracterizados por tener el objetivo de predecir valores en una escala continua, los problemas de **clasificación** son aquellos cuya variable objetivo y es discreta, finita y acotada, usualmente limitada a unos pocos valores diferentes de categoría, clase o etiqueta. Cuando el número de clases es dos se habla de clasificación binaria; si el número de clases es mayor se habla de clasificación multiclasa.

Consecuentemente, el problema definido en este TFG es un problema de aprendizaje supervisado y, en particular, de clasificación: dada una imagen craneal (entrada nueva) se desea identificar la imagen facial correspondiente (salida en la clasifica la entrada).

2.2.2. Redes Neuronales Artificiales

Las Redes Neuronales Artificiales (*Artificial Neural Networks*, ANN) [15] o simplemente Redes Neuronales son un tipo de modelos de AA constituidos por un conjunto de nodos (neuronas) conectados entre sí y organizados en capas de neuronas secuenciales. La Figura 2.4 visualiza el concepto de ANN: en este caso, un modelo compuesto por una capa de entrada (que recibe los datos de entrada), una capa oculta de neuronas y una capa de salida (que genera una salida dada una entrada).

A excepción de la capa de entrada, todas las capas de una ANN se componen de una serie de neuronas de entrada x_1, \dots, x_d , un bias w_0 , un conjunto de pesos de conexión $w_{1,1}, w_{1,2}, \dots, w_{d,m}$ y una serie de neuronas de salida y_1, \dots, y_m , donde d es el número de neuronas de la anterior capa y m es el número de neuronas de la capa actual. Cada una de las salidas es la suma de las entradas ponderadas con los pesos de conexión y el bias.

$$y_i = \sum_{j=1}^d w_{j,i} x_j + w_0 \quad (2.1)$$

En AA es usual trabajar con vectores de números como entradas y salidas

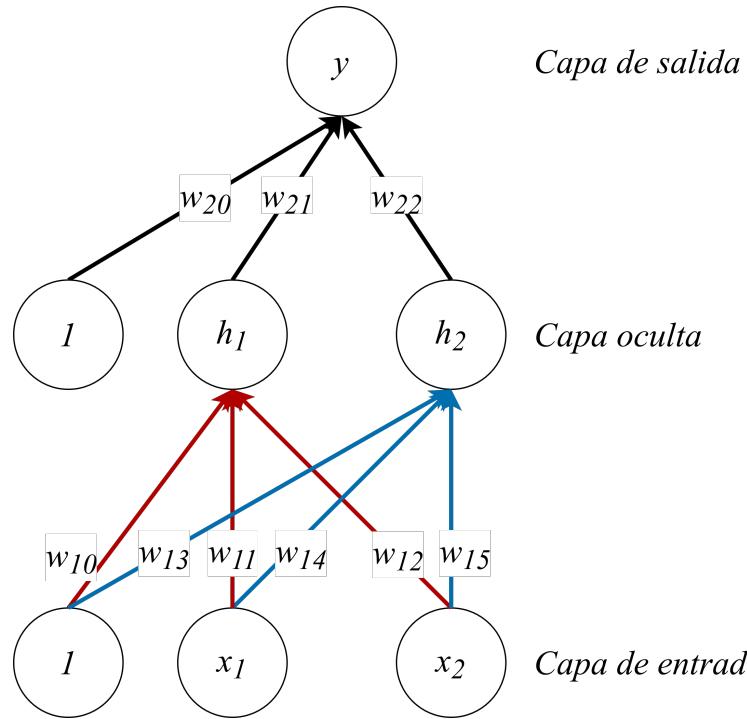


Figura 2.4: Estructura de una ANN básica con una capa de entrada (con dos neuronas), una capa oculta (también con dos neuronas) y una capa de salida (con una única neurona). Las circunferencias representan las neuronas de la red, mientras que las flechas representan la conexión entre ellas (los diferentes colores reflejan las conexiones de una capa con cada neurona de la siguiente capa). x_1, x_2 son las entradas, $w_{1i}, i = 0, \dots, 5$ son los pesos de la capa oculta y $w_{2j}, j = 0, 1, 2$ son los pesos de la capa de salida, que produce una salida y . Nótese que en cada capa se incluye una neurona correspondiente al bias, añadiéndose un peso adicional asociado a este en cada capa. Elaboración propia.

de los modelos. Se construye entonces el vector de entrada $\mathbf{x} = [1, x_1, \dots, x_d]$, extendiéndolo con un 1 para la utilización del bias, y la matriz de pesos

$$\mathbf{W} = \begin{pmatrix} w_{0,0} & w_{1,0} & \cdots & w_{0,m} \\ w_{0,1} & w_{1,1} & \cdots & w_{1,m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{d,0} & w_{d,1} & \cdots & w_{d,m} \end{pmatrix}$$

De esta forma se puede escribir la Ecuación 2.1 de la siguiente forma:

$$y = \mathbf{W}^T \mathbf{x} \quad (2.2)$$

Función de activación

Una función de activación es una función aplicada sobre una neurona para generar su salida, que será conectada con la siguiente capa de la red. Dada una función de activación $f : \mathbb{R} \rightarrow \mathbb{R}$, se actualiza la Ecuación 2.2 de la siguiente manera:

$$y = f(\mathbf{W}^T \mathbf{x}) \quad (2.3)$$

Las funciones de activación permiten introducir no linealidades en las ANN, haciendo que estas sean capaces de aprender funciones no lineales. Las más comunes son la función sigmoidal (Ecuación 2.4) y la función ReLU (Ecuación 2.5).

$$y = \frac{1}{1 + \exp(\mathbf{W}^T \mathbf{x})} \quad (2.4)$$

$$\text{ReLU} = \begin{cases} x, & x \geq 0 \\ 0, & x \leq 0 \end{cases} \quad (2.5)$$

Función de pérdida

Una función de pérdida (también referida directamente como *pérdida*) es una función diseñada para medir el rendimiento de una ANN en un problema específico. En este TFG, enfocado al problema de aprendizaje supervisado descrito ya anteriormente, se va a trabajar con dos funciones de pérdida diferentes: *Mean Squared Error* (MSE) y *Triplet Loss*. La función MSE mide el grado de diferencia entre las etiquetas reales de los datos y las etiquetas predichas por la red; se define de la siguiente forma:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (2.6)$$

Para ver la descripción de la función de pérdida Triplet Loss, consultar la Sección 2.3.2.1.

Entrenamiento de las Redes Neuronales

Las ANN están compuestas por un conjunto de capas definidas, a su vez, por matrices de pesos que conectan las entradas con las salidas. Estos pesos, denominados parámetros de una red, son variables de este tipo de modelos y deben ser ajustados para adaptarlos a cada problema concreto. Para ajustar estos pesos se debe disponer de datos sobre el problema: en general, se debe disponer de un **conjunto de datos de entrenamiento** (utilizado para ajustar los pesos al problema) y un **conjunto de datos de test** (utilizado para medir el rendimiento del modelo en un escenario

diferente al de entrenamiento). Al proceso de adaptar los parámetros de una red se le llama **entrenamiento** de la red neuronal. A las diferentes variables en la configuración del entrenamiento de una red se les denomina **hiperparámetro**.

El algoritmo de entrenamiento de una red neuronal se llama **optimizador**. Estos algoritmos utilizan la función de pérdida iterativamente durante el entrenamiento para actualizar los pesos de la red, siendo su objetivo reducir el valor de dicha función. Por regla general, a mayor función de pérdida, mayor es el cambio realizado en los pesos de la red. De cara a actualizar los pesos de la red, los datos de entrenamiento son mostrados al optimizador en **batches**, subconjuntos de datos cuyo tamaño es fijo en el entrenamiento y un hiperparámetro a definir. Los optimizadores suelen disponer de un hiperparámetro ajustable, el *learning rate*, que define el grado en el que se varían los pesos de la red cada vez que se actualizan. Otro hiperparámetro clave en el entrenamiento de una red neuronal es el número de **épocas**, número de veces que el optimizador utiliza el conjunto de datos de entrenamiento al completo (todos los batches de los que se dispone) para actualizar los pesos.

Si tras el entrenamiento de una ANN su rendimiento en el conjunto de entrenamiento es superior a su rendimiento en el conjunto de test, entonces se dice que la red tiene **overfitting**, sobreajuste a los datos de entrenamiento o que la red tiene mala **generalización**. Una de las técnicas más utilizadas para evitar el sobreajuste es la **regularización** [22], la cual consiste en limitar el valor de los pesos para obtener una distribución de pesos más regular y uniforme -de ahí el nombre de la técnica-. Esto se consigue introduciendo en la función de pérdidas términos que penalicen un modelo dados sus pesos (típicamente, un término proporcional a la suma de los valores absolutos de los pesos o la suma de los cuadrados de los mismos).

2.3. Deep Learning

DL es un campo del AA desarrollado en las dos últimas décadas basado en el aprendizaje a partir de datos utilizando múltiples niveles de representación [38, 68, 103]. Los datos de entrada son transformados sucesivamente para obtener representaciones simples con diferentes niveles de abstracción.

Las técnicas DL se materializan en las Redes Neuronales Profundas (*Deep Neural Networks*, DNN), una evolución de las ANN con una gran extensión de capas ocultas, característica por la cual reciben la denominación de *profundas*³. A mayor profundidad de tiene una DNN, mayor es su capacidad de aprendizaje (permiten aprender funciones de alta complejidad) y, por contra, mayor coste computacional tiene asociado. El desarrollo de nuevas tecnologías, como la computación basada en GPU (que permite un alto paralelismo, agilizando la ejecución de tareas costosas computacional-

³La profundidad de una ANN/DNN es el número de capas totales de la red.

mente), es un elemento clave en el reciente auge del empleo de las técnicas de DL en problemas de gran complejidad.

Por las características de los modelos de DL (recibir datos de entrada, procesarlos, ofrecer diferentes niveles de representación de los mismos), su aplicación a problemas de VC y procesado de imágenes es de especial interés, viéndose reflejado en la gran cantidad de trabajos existentes y en la calidad de los resultados obtenidos. Ejemplos de ello son el reconocimiento de dígitos manuscritos [79] o el reconocimiento de objetos en imágenes generales [116]. Este extraordinario rendimiento permite pensar que la utilización de modelos de DL en problemas de las CF (ver Sección 2.1) puede producir resultados sobresalientes. En este sentido, las Redes Neuronales Convolucionales adquieren especial importancia.

2.3.1. Redes Neuronales Convolucionales

Las Redes Neuronales Convolucionales (*Convolutional Neural Networks*, CNN o ConvNet) [69, 70] son una especialización de las DNN para trabajar con imágenes como entradas de la red. Introducen como novedad las capas convolucionales: capas compuestas de filtros (matrices cuadradas, usualmente de tamaños⁴ impares como 3×3 , 5×5 o 7×7) aplicados mediante la operación de convolución sobre la entrada de la capa.

Desde su introducción a finales del siglo pasado, cuando ya demostraron que podían superar al resto de modelos de AA, las CNN han supuesto una revolución en el AA y en el DL y son actualmente el tipo de red neuronal más utilizada [30, 132]. Esto es debido a las múltiples ventajas que presentan respecto a las ANN, entre las que se encuentran:

- En el ámbito de la CV implican una disminución importante del número de parámetros totales de la red neuronal. Para una imagen de $224 \times 224 \times 3$, solamente una de las neuronas de la primera capa de una ANN tendría asociados $224 \times 224 \times 3 = 150.528$ parámetros, mientras que en una CNN toda la primera capa convolucional con, por ejemplo, 6 filtros de 5×5 tendría asociados únicamente 456 parámetros. No solo es una ventaja tremenda a nivel computacional sino que también añade regularización al modelo.
- Tienen capacidad para procesar entradas de diferentes tamaños. Mientras que las capas FC de las ANN funcionan únicamente con un tamaño de entrada fijo, las capas convolucionales de las CNN pueden aceptar tamaños variables.
- Generan representaciones equivariantes respecto a la traslación, por lo que si un objeto se traslada en la imagen de entrada, su representación se traslada la misma distancia en la salida. Esta propiedad es de

⁴Todos los filtros de una misma capa convolucional han de tener el mismo tamaño.

especial interés en la detección de formas, bordes y texturas en toda la imagen.

Como se describía en previamente en la Sección de Redes Neuronales Artificiales (Sección 2.2.2), las capas que constituyen las ANN conectan la secuencia de neuronas de entrada con la secuencia de neuronas de salida a través de unos pesos. Las **capas convolucionales** son distintas: trabajan con volúmenes de neuronas (o píxeles) y están compuestas de neuronas organizadas en mapas de características. Los mapas de características son volúmenes tridimensionales de neuronas, estructuradas en planos de neuronas paralelos (cada plano es un filtro o kernel), como se puede ver en la Figura 2.5.

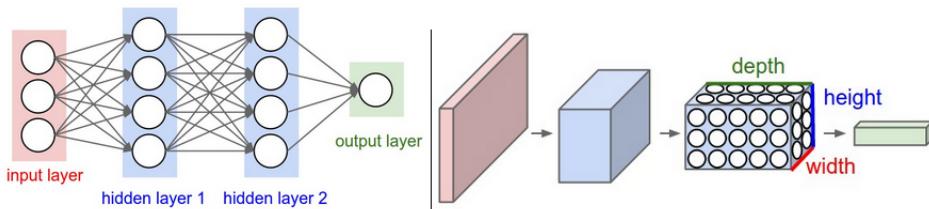


Figura 2.5: Comparativa de la estructura de las capas totalmente conectadas (izquierda, una ANN de dos capas ocultas FC) y la estructura de las capas convolucionales (derecha, una CNN con dos capas convolucionales). En ambas redes se representa cada entrada en rojo, las capas ocultas en azul y la capa de salida en verde. En el caso de la CNN la entrada (una imagen) produce como salida un volumen de neuronas. Imagen extraída de [112].

Una capa convolucional recibe un volumen de neuronas como entrada y genera un volumen de neuronas como salida, denominado mapa de activación de la capa. La salida es generada mediante la operación de convolución de cada uno de los filtros o kernels de la capa sobre el volumen de neuronas de la entrada. Cada filtro es deslizado sobre el volumen de entrada, colocando el elemento central del filtro en cada píxel o elemento que compone el volumen, aplicando la convolución y guardando el resultado en el correspondiente píxel del volumen de activación de la capa, como se puede ver en la Figura 2.6. Es frecuente aplicar una función de activación sobre el resultado producido por la convolución. En la Figura 2.7 se muestra el volumen de entrada de una capa convolucional y el mapa de activación generado por esta como salida.

Se puede comprobar como, debido a la naturaleza la operación de convolución y al tamaño de los filtros utilizados, la anchura y altura de la entrada disminuyen en el tamaño del filtro menos una unidad. De cara a evitar esta disminución de dimensionalidad es posible aplicar un *padding* [22] a los volúmenes de entrada: aumentar la dimensionalidad de dichos volúmenes antes de convolucionar los filtros. Existen diversos métodos de *padding*: re-

flejar los últimos píxeles, añadir un marco de ceros o replicar los píxeles de los bordes.

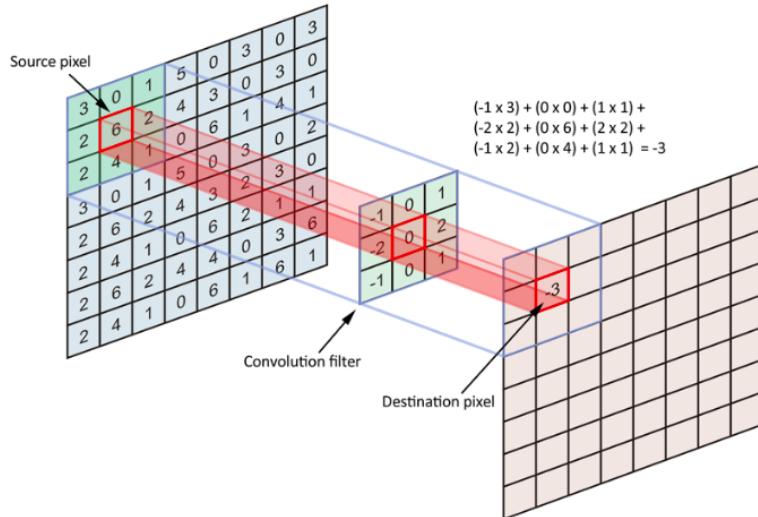


Figura 2.6: Ejemplo de convolución de un filtro de tamaño 3×3 sobre una entrada de dimensión 8×8 , produciendo una salida de dimensión 6×6 . Nótese que la reducción de dimensionalidad producida durante la operación: el filtro no puede ser aplicado sobre 1 entrada en las filas y columnas de los extremos. Imagen extraída de [32].

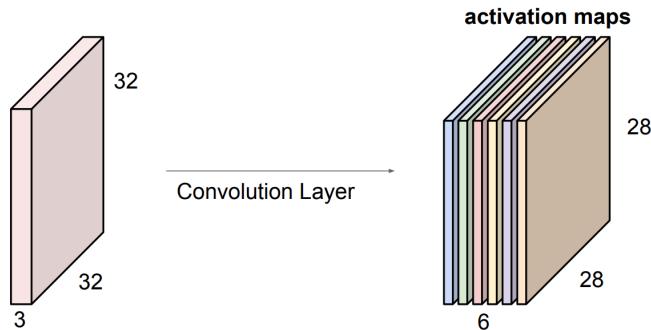


Figura 2.7: Ejemplo del funcionamiento de una capa convolucional con 6 kernels de tamaño 5×5 . De nuevo, debido a la reducción de dimensionalidad por la convolución, los 6 mapas de activación generados tienen un tamaño de 28×28 cuando el volumen de entrada tenía tamaño 32×32 . Imagen extraída de [71].

Los parámetros principales de las capas convolucionales son su profundidad (el número de filtros de la capa), el tamaño de estos (su altura y anchura) y el *padding* aplicado (si es que procede aplicar *padding*).

Las CNN muy frecuentemente incluyen otro tipo de capas más allá de las capas convolucionales [22]:

- Capas densas o totalmente conectadas. Son las utilizadas en las capas ocultas de las ANN.
- Capas de *pooling*. Este tipo de capas reducen el tamaño del volumen de entrada. Para ello, agrupan píxeles adyacentes (ventanas de píxeles) en un único píxel, reduciendo la dimensionalidad y la complejidad de la red y mejorando la generalización de esta. Aunque los dos tipos de *pooling* (ver Figura 2.8) más frecuentes son *Max Pooling* (consistente en mantener el máximo de las ventanas de píxeles) y el *Average Pooling* (consistente en mantener la media de las ventanas de píxeles), también existen otros tipos basados en diferentes operaciones, como el *L2 pooling*, que mantiene la norma L2 de las ventanas de píxeles.
- Capas de *Dropout*. Estas capas inhabilitar aleatoriamente algunas conexiones entre sus entradas y sus salidas: el grado de *dropout* se puede ajustar con el ratio de *dropout* de la capa (entre 0 y 1), la proporción de neuronas que son desconectadas. Esto permite que la siguiente capa de la red no reciba toda la información de entrada, reduciendo el sobreajuste a los patrones de los datos de entrenamiento. Es frecuente utilizarlas en conjunción con las capas densas. En la Figura 2.9 se puede ver un ejemplo de la aplicación de *dropout* a las capas ocultas (capas densas) de una ANN. En el anterior ejemplos se ha utilizado una ANN por su simplicidad, pero la aplicación de *dropout* a una CNN es exactamente igual.
- Capas de *Batch Normalization*. Estas capas aplican una normalización sobre su entrada: le restan su media y la dividen entre su desviación típica. Generalmente se utiliza tras una capa densa o una capa convolucional, antes de añadir una función de activación sobre estas capas, normalizando de forma independiente los diferentes batches. Las dos ventajas principales que ofrecen las capas de *Batch Normalization* son una mejora en la generalización de la red (debido a la aleatoriedad de las muestras de cada batch) y una aceleración del entrenamiento de la red (gracias a que evitan que haya valores de salida elevados).

2.3.2. Redes Neuronales Siamesas

Las redes neuronales siamesas (*Siamese Neural Networks*, SNNs) [98, 102] son un tipo de modelo de red neuronal en la que se utilizan dos o más sub redes neuronales idénticas (por ejemplos, una CNN replicada dos o más veces), que comparten la propia arquitectura de la red y sus pesos.

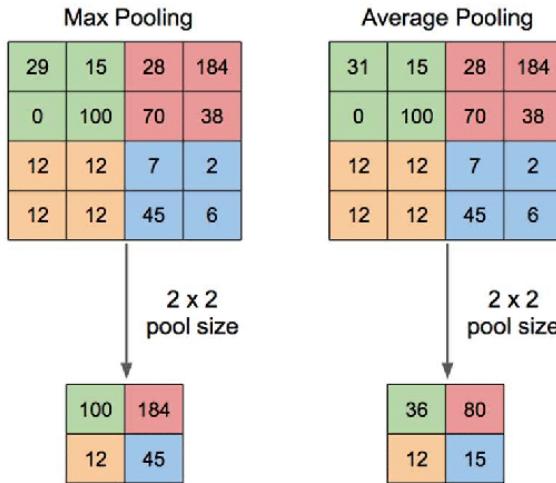


Figura 2.8: Ejemplos de capas de *pooling*, utilizando ventanas de tamaño 2; a la izquierda *Max Pooling* y a la derecha *Average Pooling*. Imagen extraída de [131].

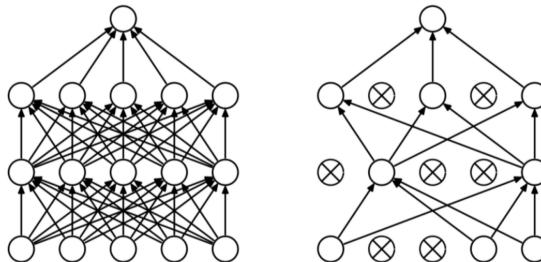


Figura 2.9: Ejemplo del empleo de *dropout* en una ANN: a la izquierda se muestra la ANN convencional y a la derecha el resultado tras aplicar *dropout* a sus dos capas ocultas. Como se puede ver, varias conexiones entre neuronas son desactivadas de tal forma que no intervengan en el resultado producido. Extraído de <https://bit.ly/3QG20eM>.

Las SNN reciben varias entradas paralelamente y cada una de ellas alimenta una de las sub redes que conforman la SNN. En la Figura 2.10 se muestra un ejemplo de SNN con tres sub redes.

Las sub redes se encargan de procesar sus respectivas entradas, generando vectores de características o *embeddings* $E(x)$ (vectores con menor dimensionalidad que la entrada pero semánticamente equivalentes) como salida; es usual referirse al bloque de sub redes como generador de *embeddings*. Por tanto, la entrada de una SNN es una secuencia de entradas, una por cada sub red neuronal, y cada una alimenta su respectiva sub red. La salida de la SNN es el conjunto de *embeddings* generados para las entradas.

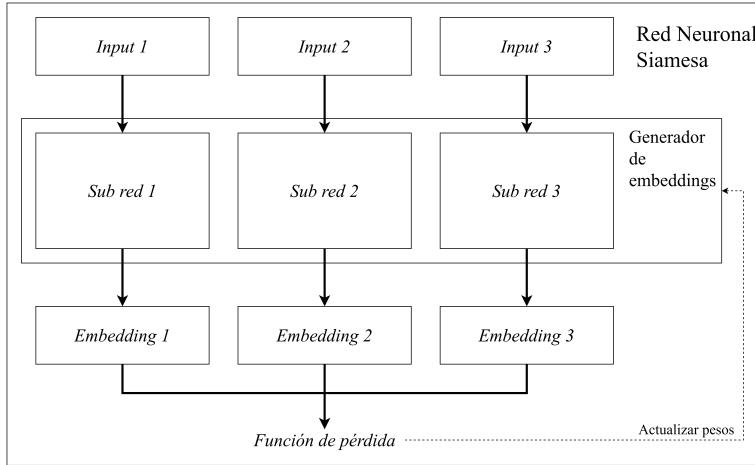


Figura 2.10: Arquitectura de una SNN con tres sub redes neuronales. La SNN recibe tres entradas y alimenta cada una de ellas a la respectiva sub red para generar el correspondiente embedding. El generador de *embeddings* es la arquitectura utilizada por las tres sub redes neuronales: los tres *embeddings* son combinados mediante la función de pérdida para actualizar los pesos del generador durante el entrenamiento. Elaboración propia.

La aplicación de este tipo de redes es, frecuentemente, diferente a la de las ANN: las SNN están diseñadas para aprender similitudes entre las entradas de la red. Durante el entrenamiento de una SNNs se busca adaptar el generador de *embeddings* para producir *embeddings* similares para entradas similares y *embeddings* diferentes para entradas diferentes. La similitud entre *embeddings* es medida por una función de distancia, específica para cada problema, aunque es común utilizar la distancia L2. La distancia L2 (o distancia euclídea) entre dos *embeddings* es definida como:

$$d(p, q) = \sqrt{\sum_{i=0}^d (p_i - q_i)^2} \quad (2.7)$$

Las SNN son generalmente utilizadas en problemas con baja disponibilidad de datos, gracias a que aprenden similitudes entre datos [55, 105]. En vez de aprender a clasificar los datos en diferentes clases, tarea difícil en situaciones en las que se dispone de uno o pocos ejemplos para cada clase (ya que es muy probable un sobreajuste en el entrenamiento), las SNN aprenden a diferenciar los aspectos que caracterizan los datos. Permiten obtener, así, un vector de características que los represente, pudiendo comparar estos vectores para establecer el grado de diferencia entre los datos de entrada.

2.3.2.1. Función de pérdida Triplet Loss

La función de pérdida *Triplet Loss* (TL) es una función de pérdida diseñada para el entrenamiento de SNNs con tres sub redes [19, 105, 106]. La TL busca minimizar la distancia intra-clase entre los *embeddings* generados a la vez maximizar la distancia inter-clase entre los *embeddings*.

Para conseguir lo anterior, la SNN es entrenada utilizando tripletas de ejemplos de la forma (A, P, N) , donde A es el *ancla*, P es el *positivo* y N es el *negativo*. El ejemplo ancla es el ejemplo principal o referencia de la tripleta, el ejemplo positivo es un ejemplo de la misma clase que el ancla y el ejemplo negativo es un ejemplo de cualquier otra clase diferente a la del ancla.

El objetivo de la TL es que la distancia entre *embeddings* semánticamente similares sea menor que la distancia entre *embeddings* semánticamente diferentes, dentro de un margen de error α . Es decir, el objetivo es que las tripletas satisfagan la Ecuación 2.8, d es la función distancia y α es el margen (cuyo valor es determinado experimentalmente). Durante el entrenamiento, esto se traduce en separar fuera del margen α la distancia entre los *embeddings* del ancla y del positivo y la distancia entre los *embeddings* del ancla y del negativo. Con todo, se llega a la función de pérdida mostrada en la Ecuación 2.9.

$$d(E(A), E(P)) + \alpha < d(E(A), E(N)) \quad (2.8)$$

$$TL(A, P, N) = \max\left(0, d(E(A), E(P)) - d(E(A), E(N)) + \alpha\right) \quad (2.9)$$

Para entrenar una SNN, esta es iterativamente alimentada con las tripletas de ejemplos de entrenamiento, generando sus vectores de características y calculando la TL sobre estos para actualizar los pesos del generador de *embeddings* de la SNN⁵. Se considera que una tripleta está correctamente clasificada si satisface la Ecuación 2.8.

2.3.2.2. Redes Siamesas con salida sigmoidal

Las SNN también pueden ser utilizadas para producir una salida entre 0 y 1, de tal forma que las entradas se consideran similares si la salida es 1 y diferentes si la salida es 0. Una salida categórica (0 o 1) resulta menos práctica que una salida continua (en el intervalo [0, 1]), ya que no permite diferenciar grados de similitud entre las entradas. Por esta razón, es posible combinar los *embeddings* generados (por ejemplo, concatenándolos), emplear un modelo de red neuronal sobre ello y producir una salida sigmoidal [43, 44, 59]. La Figura 2.11 ilustra este tipo de arquitecturas.

⁵Recordar que la SNN está formada por una única sub red replicada tres veces.

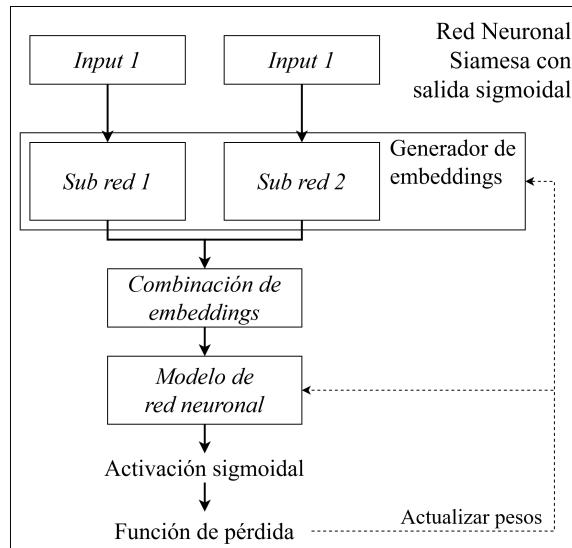


Figura 2.11: Arquitectura de una SNN con dos sub redes neuronales y salida sigmoidal. La SNN recibe dos entradas y alimenta cada una de ellas a la respectiva sub red para generar el correspondiente embedding, que posteriormente son combinados y pasados a un modelo de red neuronal que produce una salida sigmoidal. La salida de este tipo de arquitecturas puede interpretarse como similar o no similar en función de como de cercana es a 1 o a 0, respectivamente. Elaboración propia.

2.3.3. Otros conceptos relevantes

2.3.3.1. One Shot Learning

Los problemas *One Shot Learning* constituyen un subtipo de problemas de aprendizaje supervisado que se caracterizan por la baja disponibilidad de datos [56]. Son problemas que se desarrollan en escenarios donde se tiene uno o muy pocos ejemplos de cada clase: *One Shot Learning* es un enfoque para resolver tareas de aprendizaje supervisado con datos limitados utilizando conocimientos previos.

Por ello, dentro de los problemas de AA, son los más parecidos al aprendizaje del cerebro humano, ya que las personas son capaces de aprender patrones a partir de muy pocos ejemplos: un humano puede aprender a reconocer caballos habiendo visto una única imagen de un caballo. En particular, en el ámbito del DL suponen un reto importante en el entrenamiento de los modelos debido a la dependencia de este proceso en una gran cantidad de datos.

El problema que se afronta en este trabajo es un problema de *One Shot Learning* debido a que se dispone únicamente de un ejemplo de cada clase: para cada individuo del conjunto de datos se dispone de un modelo 3D

craneal y, en algunos casos, una imagen facial.

2.3.3.2. Transfer learning y Fine-tuning

Uno de los mayores inconvenientes de las CNNs es la necesidad de disponer de grandes cantidades de datos para alimentar la red neuronal durante el entrenamiento. A mayor disponibilidad de datos y a mayor varianza de estos, mejor rendimiento obtendrá la red, ya que estará expuesta a una mayor cantidad de casos diferentes durante el entrenamiento. Un aspecto fundamental en el entrenamiento de una CNN es, por tanto, la disponibilidad de datos de entrenamiento.

Transfer Learning es una técnica de DL que consiste en emplear un modelo entrenado para un problema concreto para otro problema distinto. En otras palabras, *Transfer Learning* consiste en utilizar una CNN existente y preentrenada en otro conjunto de datos diferentes (ya sea uno genérico o uno parecido al disponible) y reentrenarla para un nuevo problema. Esto permite utilizar el conocimiento adquirido previamente por la red y, dado el coste computacional y la necesidad de grandes conjuntos de datos para entrenar desde cero una CNN, hace sea la técnica generalizada a la hora de resolver problemas (especialmente los problemas *One Shot Learning*) mediante CNNs.

Uno de los métodos más comunes de *Transfer Learning* es el *Fine-tuning*. Esta técnica consiste en congelar (durante el entrenamiento, congelar una capa implica que los pesos de esta no son modificados ni actualizados) partes del modelo preentrenado y reentrenar únicamente las capas finales (las más profundas) en los datos disponibles. El hecho de no reentrenar las capas más superficiales (las primeras de la red) es debido a que estas aprenden a extraer características más generales, mientras que las capas más profundas se especializan en características más finas propias de cada problema concreto.

2.3.3.3. Data augmentation

Otra técnica cuyo objetivo es aliviar el problema de la disponibilidad de datos consiste en generar datos realísticos⁶ y alimentar la red neuronal tanto con los datos reales como con los generados de esta manera. *Data augmentation* consiste en, a partir de los datos de entrenamiento disponibles, crear nuevos ejemplos realísticos y adecuados para el problema. La última condición es importante, ya que generar datos incompatibles con el problema provocaría que el rendimiento de la red disminuiría. Por ejemplo, si se el entrenamiento se realiza sobre imágenes faciales y se aplica *data augmentation* generando nuevas imágenes aplicando desplazamientos aleatorios sobre

⁶Generar datos realísticos se refiere a generar datos que sean lo más parecidos en naturaleza a los datos reales sin que por ello sean copias exactas de valores. Se trata de insertar variaciones sobre datos existentes para producir nuevas instancias de datos tales que un experto no pueda distinguir entre ellas y las reales.

las originales de tal forma que desplazan la mitad de la cara fuera de las imágenes, entonces la red recibirá ejemplos donde solo aparezca parcialmente una cara (por ejemplo, una imagen en la que se ve solo la barbilla) y su rendimiento será peor.

En la Figura 2.12 se pueden ver dos ejemplos de *data augmentation* de imágenes de caras de personas; uno de ellos es adecuado mientras que el otro genera una imagen no utilizable.



Figura 2.12: Ejemplos de generación de nuevas imágenes faciales a partir de una original. La imagen aumentada correctamente ha sido generada mediante modificaciones aleatorias del contraste, del brillo y de la saturación tras un volteo horizontal. La imagen aumentada incorrectamente ha sido generada aplicando un desplazamiento demasiado grande sobre la imagen original y no es un ejemplo adecuado de *data augmentation*. Elaboración propia.

2.4. Visión por Computador

La VC existe desde la década de 1960 [114], aunque es en las últimas décadas cuando ha experimentado un crecimiento notable. Comprende las técnicas relacionadas con el procesamiento y análisis de imágenes digitales y la extracción de información de las mismas para su posterior utilización. Las numerosas aplicaciones en el mundo real de la VC varían desde la detección de puntos clave en imágenes [136] hasta el registrado de imágenes pre y post operatorio en Medicina [67] o el reconocimiento óptico de caracteres [22]. Un campo importante en la VC es la extracción de características en imágenes (simplificar las imágenes a vectores numéricos que recojan la información presente en ellas) y la clasificación de estas en categorías (identificación de formas en imágenes, por ejemplo, determinar si una imagen contiene un gato o no).

La VC y el AA siempre han estado estrechamente ligados ya que muchos procedimientos de VC para procesado de imágenes y reconocimiento de patrones se apoyan en técnicas de AA. Con la expansión de DL, los métodos

de VC han ahondado en el empleo de este tipo de modelos. Es frecuente la utilización en tareas de extracción de características en imágenes y su clasificación de CNNs [65, 76, 72, 35].

Otro aspecto importante en la VC es la representación de la información a través de modelos, sobre todo en lo que refiere a imágenes 3D o información tridimensional. Una de las más recurridas representaciones es la conocida como **mallas poligonales**. Una malla poligonal (también será referido como malla u objeto 3D) [85] es una representación tridimensional de un objeto geométrico, formada por un conjunto de vértices y un conjunto de formas poligonales (usualmente triángulos) que unen dichos vértices. Los vértices de las mallas están definidos por coordenadas tridimensionales; las formas poligonales por índices enteros que apuntan a los índices en la lista de vértices de los vértices que conforman el polígono.

El *bounding box* de una malla es el menor hexaedro que contiene dicha malla, como se puede ver en la Figura 2.13.

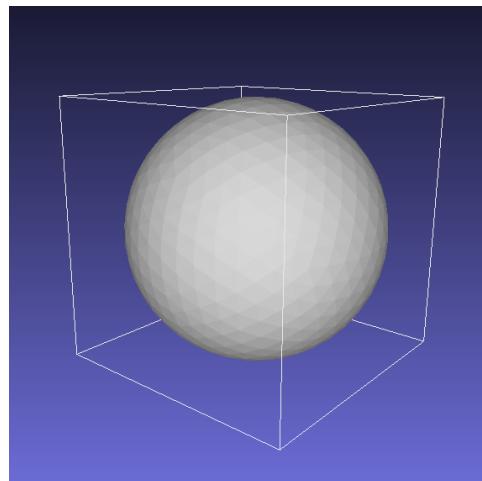


Figura 2.13: *Bounding box* de una esfera. Elaboración propia.

Se dice que dos mallas están **alineadas entre sí** si comparten el mismo sistema de coordenadas. Si dos mallas no están alineadas, mediante transformaciones geométricas se pueden alinear: traslaciones, rotaciones y escalados.

Capítulo 3

Estado del arte

Debido a que el problema abordado en este trabajo representa un reto recientemente planteado en la comunidad investigadora, la literatura concerniente al mismo se puede considerar relativamente escasa. Para el estudio de los trabajos relacionados se han consultado fuentes bibliográficas como *Scopus* o *IEEE Xplore*, incluyendo los trabajos científicos publicados en revistas y congresos con revisiones a pares. A continuación se presenta el resultado de este estudio bibliográfico, incluyendo estudios desde la clasificación de imágenes con técnicas de DL hasta las posibles alternativas publicadas abordando el problema concreto a resolver en este TFG.

3.1. La clasificación de imágenes con métodos de Deep Learning

Actualmente las técnicas de DL conforman el estado del arte en clasificación de imágenes, claramente superando a las técnicas clásicas de AA basadas en la extracción de características manual para su posterior clasificación [73, 74] y experimentando un gran crecimiento en la última década, como se puede comprobar en la Figura 3.1.

Los métodos de DL actuales para la resolución de problemas concretos se basan en *transfer learning* y en el *fine-tuning* de redes preentrenadas. El estado del arte se refiere, por tanto, a las arquitecturas de redes empleadas y no a la estructura interna (organización de capas, etc.) de estas. Dicho esto, existen múltiples arquitecturas y no se puede destacar una en concreto fuera de un contexto específico. Esto se debe a que diferentes problemas pueden requerir el empleo de diferentes redes preentrenadas.

No obstante, sí que destacan en la clasificación de imágenes las CNNs por encima de otro tipo de redes (como las ANN), gracias al aprovechamiento que hacen de la naturaleza de las imágenes [46] y la eficiencia introducida en el proceso de aprendizaje. Algunos autores coinciden en destacar dentro del estado del arte ciertas arquitecturas, las cuáles se listan en la Tabla 3.1.

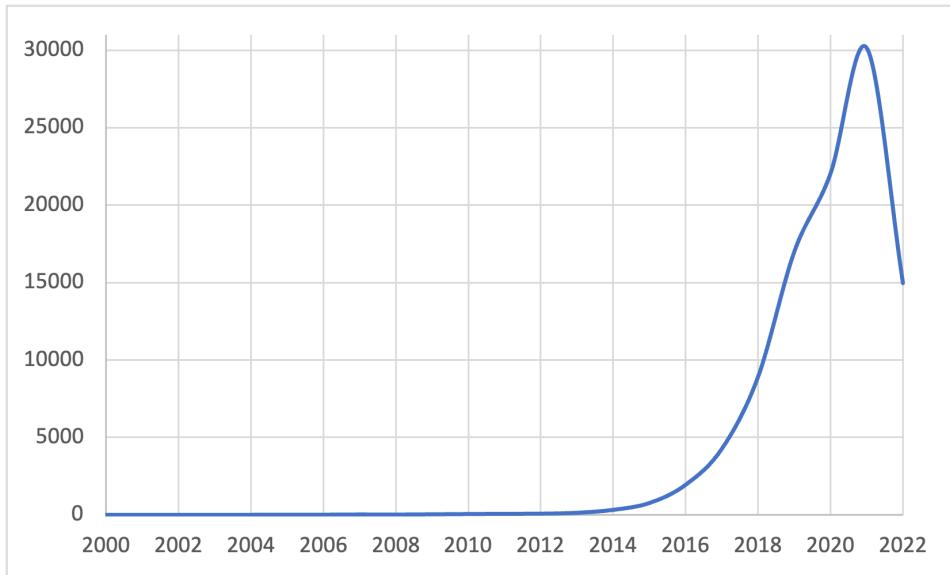


Figura 3.1: Número de artículos publicados por año en el ámbito del DL y las imágenes desde 2000. En 2012 el número de publicaciones comienza a crecer gracias al impulso recibido tras la publicación de *AlexNet* [66], que superaba a las técnicas clásicas en el dataset *ImageNet*. Imagen elaborada en *Scopus* mediante la consulta *TITLE-ABS-KEY (deep AND learning AND image)* el 1 de Julio de 2022.

En la Figura 3.2 se puede observar las citas a éstas redes en la bibliografía reciente.

El lector interesado puede obtener más detalles a través de la lectura de algunos de los trabajos de revisión de bibliografía publicados recientemente [46, 58, 62]. No obstante, no se profundiza más en este aspecto debido a que la casi totalidad de los trabajos que se encuentran en la literatura no coinciden con el reto propuesto en este trabajo; esto es así dado que la inmensa mayoría de los trabajos publicados se centran en el etiquetado de imágenes y no en el emparejado de las mismas.

3.2. Reconocimiento facial mediante Deep Learning

El reconocimiento facial ha atraído a la comunidad investigadora en los últimos años dadas las múltiples aplicaciones que tiene: seguridad, seguimiento, medicina o investigación forense, por citar algunos casos [5]. El primer intento registrado de identificación facial de un sujeto mediante una fotografía facial data de 1871 en juzgado británico [90]. En 1964 se realizó el primer estudio de identificación facial mediante sistemas computarizados

| Modelo | Ref. | Descripción | Citas recientes |
|--------------|-------|--|-----------------------|
| Inception v3 | [113] | Escala la red neuronal usando convoluciones factorizadas y regularizaciones agresivas. | [12, 57, 82, 95, 101] |
| VGG | [110] | Redes convolucionales con más capas usando filtros de pequeña dimensión. | [1, 29, 48, 60, 125] |
| ResNet | [45] | Reformulación de las capas para el aprendizaje de funciones residuo con respecto a las capas de entrada. | [6, 31, 33, 63, 75] |
| EfficientNet | [116] | Método de escalado de redes en ancho, profundidad y resolución de manera eficiente. | [2, 11, 13, 128, 130] |

Tabla 3.1: Principales diseños de redes neuronales profundas utilizadas en la clasificación de imágenes.

[17]. En 2014, Facebook presentó Deepface para el reconocimiento facial, indicando una exactitud superior al 97 % [115]. Desde entonces, el DL se ha considerado la técnica más apropiada para la clasificación de imágenes, y más específicamente, las imágenes faciales.

El reconocimiento facial incorpora varios componentes [97]: i) un módulo detector de cara -para localizar la cara en una imagen o video-, ii) un módulo de procesado de la zona de la cara -alineando y normalizando la parte de la imagen-, y iii) el módulo de reconocimiento facial propiamente dicho.

El preprocessado de las imágenes faciales se puede clasificar en *one-to-many* o *many-to-one* [129]. En el preprocessado *one-to-many* se intenta enriquecer el data set para obtener una mayor variabilidad de datos en el entrenamiento y en la validación, bien sea por medio de modelado 2D de las caras [135], el modelado 3D de las éstas [99], *data augmentation* [126]. Por contra, el preprocessado *many-to-one* genera nuevas representaciones del dominio partiendo de las ya existentes e introduciendo un patrón de ruido; para este preprocessado se suelen utilizar *AutoEncoders* [61] o *Generative Adversarial Networks* (GANs) [28].

El reconocimiento facial incluye la extracción de características profundas, la selección de la función de pérdida y el emparejado de caras por medio de las características profundas [129]. En la extracción de características profundas se utilizan, típicamente, las CNN y en especial, además las ya mencionadas anteriormente Inception v3 [113], VGG [110], ResNet [45] o EfficientNet [116], se puede contar a AlexNet [66], FaceNet [105], GoogleNet

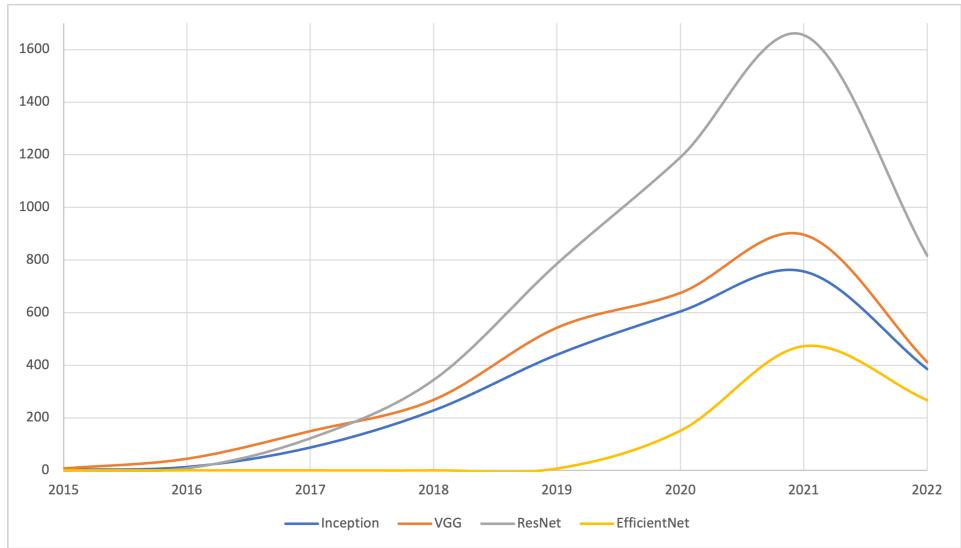


Figura 3.2: Evolución de las publicaciones en los últimos años para algunos de los diseños de redes convolucionales más utilizadas. Obtenido consultando el correspondiente modelo conjuntamente con el término *convolutional* (consulta *TITLE-ABS-KEY (modelo AND convolutional)* realizada a través de *Scopus* el 1 de Julio de 2022).

[113] o SENet [50].

La selección de la función de pérdida origina una gran variabilidad de soluciones y de rendimientos. Así como *softmax* es la más comúnmente utilizada por su capacidad de focalizar en la separabilidad de los atributos extraídos, no está claro que sea la mejor opción para el reconocimiento de caras debido a la variabilidad inherente al problema. Funciones de pérdida basada en la distancia euclídea (e.g., [66]), en el margen angular/coseno [127] y las variaciones publicadas sobre la función softmax [93, 96] son las propuestas más habituales en el ámbito de reconocimiento de caras. Cabe la pena destacar dentro de las funciones de pérdida basadas en la distancia euclídea, aquellas basadas en tripleta (*Triplet Loss*, introducida conjuntamente con FaceNet [105, 129]) que consideran la diferencia relativa de las distancias entre pares de caras.

Finalmente, el emparejado de características profundas se puede dividir en verificación facial [20] o en identificación facial [21]. En este caso, se trata de la metodología para determinar el resultado de una comparación como verdadera o falsa. Existen diversas aproximaciones, la mayor parte de ellas específicas del problema en concreto.

3.3. Identificación humana directa basada en imágenes digitales faciales y craneales

Como ya se ha mencionado, el problema de la identificación de personas basado en el emparejado de imágenes digitales faciales y craneales es de muy reciente planteamiento. Es por ello que el número de artículos con soluciones al mismo es muy reducido. En concreto, se han encontrado solo dos artículos revisados que aborden el problema de IDH utilizando imágenes faciales [88, 111]. En ambos estudios se propone la utilización de un modelo de Aprendizaje de Transformaciones para aprender una transformación sobre imágenes que extraiga vectores de características. Esta transformación es aplicada a la par sobre imágenes faciales y craneales, tal que las identificaciones son realizadas comparando los vectores de características de los cráneos con los de las imágenes faciales de los candidatos.

En [88] se propone una solución basada en dos transformaciones, T_f y T_s , que generan representaciones Z_f y Z_s de las imágenes faciales y craneales de entrada, X_f y X_s , respectivamente. Ambas transformaciones son aprendidas en función de los datos de entrenamiento mediante dos algoritmos de aprendizaje, uno de ellos no supervisado y el otro supervisado. El *algoritmo de aprendizaje no supervisado* consiste en el aprendizaje de las transformaciones T_f y T_s sobre las imágenes faciales y craneales por separado, pudiendo utilizarlas para calcular los vectores de características de las imágenes y utilizar la distancia Euclídea entre ellos para establecer identificaciones. El *algoritmo de aprendizaje supervisado* parte de las transformaciones aprendidas mediante el anterior algoritmo y añade un clasificador perceptrón con pesos W entrenables que permite decidir cuando dos imágenes (una facial y otra craneal) pertenecen a la misma persona: a partir de imágenes faciales y craneales correspondientes se reentrenan T_f y T_s y se entrena W .

Por otro lado, en [111] se utiliza, como novedad, una única transformación T que, dadas las imágenes craneales y faciales X_f y X_s de entrada, genera sus respectivas representaciones A_s y A_d . El objetivo de esta transformación compartida es proyectar las representaciones A_s y A_d al mismo espacio, mientras se minimizan las distancias intra clase (las distancias entre A_s y A_d de una misma persona son pequeñas). Para ello se utiliza un *algoritmo de aprendizaje supervisado* que optimiza la transformación T y la distancia entre los vectores de características de las imágenes faciales y craneales. En ambas publicaciones se miden los resultados sobre un conjunto de test, obteniendo una *accuracy* de 37.1 % y 42.9 %, respectivamente.

Adicionalmente, se ha encontrado una publicación no revisada por pares que utiliza una red convolucional como parte del proceso de identificación imágenes 2D faciales y craneales [40]. Este artículo no afronta el mismo problema sobre el que se trabaja en este TFG, ya que no utiliza un modelo de DL para ofrecer una clasificación directa de las imágenes de entrada. No

3.3. Identificación humana directa basada en imágenes digitales faciales y craneales

obstante, se ha analizado como un estudio en un campo similar al que se trabaja en este TFG. El método utilizado en este artículo consiste en seis pasos bien diferenciados: i) extracción de características de los pares de imágenes de entrada (una facial y otra craneal), ii) mejora de las características (por ejemplo, eliminando bordes), iii) extracción de puntos ROI, iv) modelado de landmarks mediante una red convolucional, v) superposición craneofacial y vi) identificación de la persona.

Capítulo 4

Planificación e Implementación

En este capítulo se explica cómo se ha llevado a cabo este proyecto, desglosando primero la planificación realizada con los costes asociados, describiendo el diseño del proyecto en términos de tareas implementadas, y finalizando describiendo el entorno de ejecución.

4.1. Planificación temporal y económica

El presente TFG se ha desarrollado durante el segundo cuatrimestre del curso 2021-2022, iniciando los trabajos en Febrero de 2022 y finalizando para la convocatoria de Septiembre de 2022. La planificación inicial del TFG se visualiza en la Figura 4.1, incluyendo las siguientes tareas:

- Estudio de la literatura relativa: búsquedas bibliográficas y procesado de la misma.
- Generación de las imágenes 2D craneales: partiendo de los modelos 3D, generar el conjunto de imágenes 2D con diferentes proyecciones y perspectivas de los cráneos. Implementación de los códigos necesarios.
- Diseño de las arquitecturas de DL tras el análisis de la literatura: selección de los modelos DL más prometedores y su adaptación al problema. Implementación de los códigos necesarios.
- Diseño de la experimentación: selección del protocolo de validación más adecuado, las funciones de pérdida a utilizar, los métodos de evaluación de los modelos y el conjunto de hiperparámetros. Implementación de los códigos necesarios.

- Experimentación y análisis: implementación de los scripts para ejecución en las máquinas del CPD de la Universidad de Granada. Análisis de los resultados obtenidos y extracción de conclusiones.
- Desarrollo de la memoria: desarrollo del presente documento.

| Actividad | Fe | Mr | Ab | My | Ju |
|------------------------------------|----|----|----|----|----|
| Estudio de la literatura | | | | | |
| Generación de modelos craneales 2D | | X | | | |
| Diseño de arquitecturas DL | | | X | | |
| Diseño de la experimentación | | | | X | |
| Experimentación y análisis | | | | X | |
| Desarrollo de la memoria | | | | | X |

Figura 4.1: Diagrama de Gantt con la planificación inicial del TFG.

Cabe indicar que el desarrollo de TFG se apartó de la planificación inicial en varios puntos y debido a razones diversas. Por un lado, la etapa de *Generación de los modelos craneales 2D* ocupó más tiempo del inicialmente planteado debido a la complejidad de las proyecciones y perspectivas implementadas, así como un retraso en la disponibilidad de la totalidad de los modelos.

Por otra parte, las tareas relacionadas con el *Diseño de las arquitecturas de DL*, *Diseño de la experimentación* y *Experimentación y análisis* también conllevaron más tiempo del inicialmente previsto. Así como en la primera se demoró más en el propio diseño de las arquitecturas, en la segunda se cometieron algunos errores que, a la larga, extendieron más la fase de Experimentación y análisis. Con todo ello, el desarrollo del TFG se llevó a cabo según el gráfico de Gantt reflejado en la Figura 4.2. Cabe destacar, igualmente, que el estado del arte también fue más extenso, al contrario de lo inicialmente planificado.

| Actividad | Fe | Mr | Ab | My | Ju | Jl | Ag |
|------------------------------------|----|----|----|----|----|----|----|
| Estudio de la literatura | | | | | | | |
| Generación de modelos craneales 2D | | X | | | | | |
| Diseño de arquitecturas DL | | | | | | | |
| Diseño de la experimentación | | | | | | | |
| Experimentación y análisis | | | | X | | | |
| Desarrollo de la memoria | | | | | | X | |

Figura 4.2: Diagrama de Gantt con la planificación real del TFG.

El presupuesto de este trabajo se ha realizado asumiendo su desarrollo por Investigador Senior o por un Responsable de I+D de una empresa tecnológica. Por un lado se contempla el coste salarial y por otro lado el coste de amortización de equipos.

Para la parte salarial se ha fijado un salario de 35 €/h, asumiendo una jornada parcial de 4 horas diarias. La planificación inicial, como se ha indicado anteriormente, se extiende por 120 días. Consecuentemente, el presupuesto inicial por mano de obra sería de 16.800 €. También como se ha indicado, el tiempo de desarrollo real ha sido de 219 días, por lo que el presupuesto real asciende a 30.660 €.

Por otra parte, se asume también los costes derivados de la amortización de los equipos informáticos, tanto de usuario como de cómputo. Como equipo informático de usuario se ha considerado un ordenador personal (ver Sección 4.3), con un coste de 850 €. Como equipos informáticos de cómputo se ha considerado el coste de uno de los equipos del CPD Santa Lucía indicado en la sección antes referenciada, presupuestando un equipo de estas características en 12.000 €.

El equipo informático de usuario se ha utilizado durante todo el tiempo de proyecto (120 o 219 días), con una vida útil de 5 años (1.825 días). Por otro lado, se ha planificado un uso de equipo informático de cómputo de 20 días en el supuesto inicial, 30 días en el desarrollo real. También se ha contemplado una amortización en 5 años (1.825 días).

Consecuentemente, los costes imputables al desarrollo del TFG por el material informático ascendería a 55,89 € debido a los equipos informáticos de usuario y 131,51 € debido a los equipos informáticos de computación en la planificación inicial. En el desarrollo real, los costes ascienden a 102,00 € y 197,26 €, respectivamente para cada concepto.

En resumen, el presupuesto inicial de este TFG asciende a 16.987,40 €, mientras que el presupuesto final es de 30.959,26 €.

4.2. Diseño e implementación

El presente TFG está orientado al estudio de viabilidad de modelos de DL para la IDH a partir de fotografías de cráneos y caras de personas que, en principio, figuran como desaparecidas en los registros de denuncias de desapariciones tanto nacional como internacional. Por tanto, el objetivo no es diseñar y crear un software para terceros.

No obstante, esto no quita que haya sido necesario cuidar el diseño del software desarrollado dada la proporción del proyecto. Para este fin se ha diseñado la secuencia de etapas indicadas en la sección anterior, la cuál se refleja más detallada y con los solapamientos en el tiempo de la Figura 4.3. Se ha diseñado una aplicación con doble finalidad: por un lado, generar imágenes 2D craneales a partir de modelos 3D; por otro lado, construir, entrenar, validar y realizar el test de los modelos de DL. Se han diseñado las herramientas pertinentes para guardar y analizar resultados, integrándolas en la anterior aplicación. Todo ello ha sido llevado a cabo mediante una metodología de desarrollo de software en cascada [91] (ver Figura 4.4), con

sus respectivas fases de análisis, diseño, codificación, pruebas, puesta en funcionamiento y posterior documentación y mantenimiento.

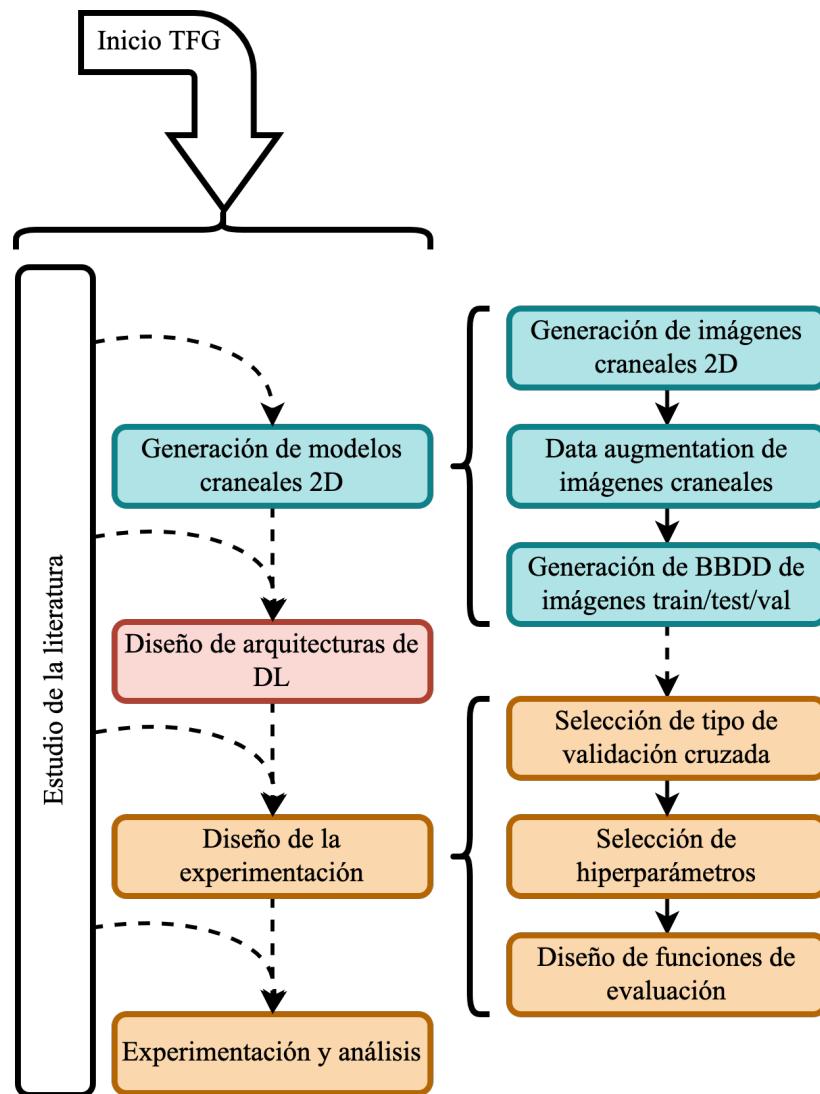


Figura 4.3: Diseño e implementación del presente TFG.

El modelo en cascada puro no es muy frecuentemente utilizado, ya que su uso implicaría un previo y completo conocimiento de los requisitos, la invariabilidad e incondicionalidad de los mismos y la absoluta ausencia de errores en etapas siguientes del desarrollo. El modelo en cascada puro es, por tanto, solo viable y únicamente empleado desarrollos de pequeños sistemas y aplicaciones. En cambio, este proyecto, de un espectro más amplio, requiere una realimentación entre la secuencia de etapas que permite una posible modificación de unas tras el desarrollo de otras siguientes. El modelo en cas-

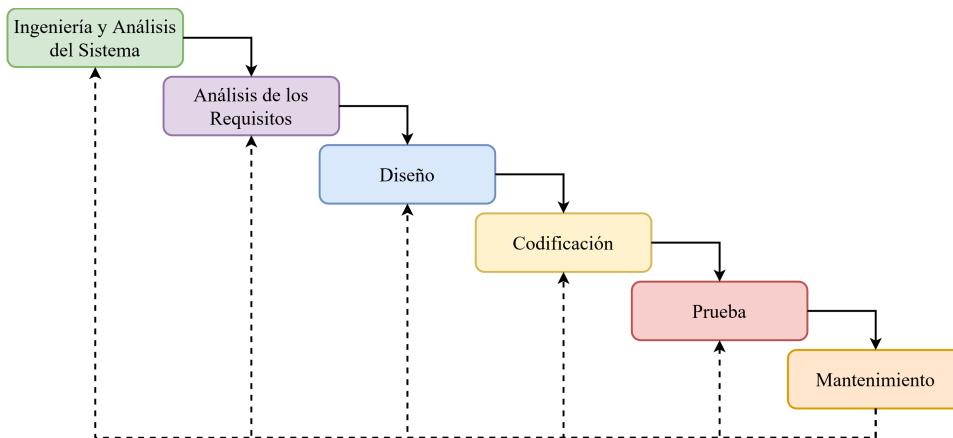


Figura 4.4: Metodología de desarrollo en cascada empleada en este proyecto.

cada empleado es uno de los más utilizados, por su eficacia y simplicidad, en software de medianas dimensiones, como es el caso. Permite producir realimentaciones entre etapas, permitiendo el desarrollo de productos software en los que hay ciertas incertidumbres, cambios o evoluciones durante su ciclo de vida. Así por ejemplo, tras la especificación de los requisitos, se puede pasar al diseño del sistema, pero si durante esta siguiente fase se debiesen realizar ajustes en los requisitos previos (aunque sean mínimos), ya sea por errores detectados, ambigüedades o bien porque los propios requisitos han cambiado o evolucionado, se retorna a la etapa anterior para realizar los pertinentes reajustes y luego continuar nuevamente con el diseño.

Este modelo se considera como adecuado para el desarrollo del presente trabajo por los siguientes motivos:

- El proyecto presenta una alta rigidez, ya que no se esperaban cambios sustanciales en su desarrollo.
- Los requisitos y objetivos del proyecto estaban bien especificados desde el comienzo y su definición era clara.
- El modelo permite la modificación de etapas previas ante la posible aparición de imprevistos y la detección de errores en etapas posteriores del desarrollo

El análisis de requisitos se realizó desde el punto de vista de la Inteligencia Artificial y de las Ciencias Forenses, en conjunción con los tutores de este TFG.

La implementación inicial de este TFG ha sido desarrollada utilizando cuadernos de Google Collaboratory [39], permitiendo diseñar las diferentes etapas y subetapas mostradas en la Figura 4.3. Los cuadernos anteriores

sirvieron para bosquejar el desarrollo final de los scripts en Python para la ejecución de la experimentación en la infraestructura de computación de la Universidad de Granada. Estos scripts están disponibles en https://github.com/MarioVillar/TFG_IDH. El contenido de estos archivos es el siguiente:

- Directorio **entrenamiento_modelos_dl**. Contiene los archivos relativos al entrenamiento y evaluación de los modelos de DL.
 - Archivo **macros.py**. Macros y variables comunes a las diferentes ejecuciones.
 - Archivo **distance_functions.py**. Definición de distancias entre imágenes.
 - Archivo **image_preprocessing_functions.py**. Funciones de preprocesado de imágenes.
 - Archivo **read_pkl_info.py**. Leer información guardada en disco durante el entrenamiento de los modelos.
 - Archivo **test_functions.py**. Funciones relativas a la evaluación de los modelos.
 - Archivo **threshold_detection_functions.py**. Funciones relativas a la detección del umbral de decisión de los modelos.
 - Archivo **utilities.py**. Diversas utilidades: funciones para mostrar gráficas, salvar resultados a disco, etc.
 - Archivo **main_hold_out_snn_tl_basic.py**. Main de ejecución de Hold-out sobre los modelos SNNTLBASIC.
 - Archivo **main_hold_out_snn_tl_sel.py**. Main de ejecución de Hold-out sobre los modelos SNNTLSEL.
 - Archivo **main_hold_out_snn_tlc.py**. Main de ejecución de Hold-out sobre los modelos SNNTLC.
 - Archivo **main_hold_out_snn_tl_online.py**. Main de ejecución de Hold-out sobre los modelos SNNTLONLINE.
 - Archivo **main_hold_out_snn_sigmoid.py**. Main de ejecución de Hold-out sobre los modelos SNNSIGMOID.
 - Archivo **main_cv_snn_tl_basic.py**. Main de ejecución de Validación Cruzada sobre los modelos SNNTLBASIC seleccionados en Hold-out.
 - Archivo **main_cv_snn_tl_sel.py**. Main de ejecución de Validación Cruzada sobre los modelos SNNTLSEL seleccionados en Hold-out.
 - Archivo **main_cv_snn_tlc.py**. Main de ejecución de Validación Cruzada sobre los modelos SNNTLC seleccionados en Hold-out.

- Archivo **main_cv_snn_tl_online.py**. Main de ejecución de Validación Cruzada sobre los modelos SNNTLONLINE seleccionados en Hold-out.
- Archivo **main_cv_snn_sigmoid.py**. Main de ejecución de Validación Cruzada sobre los modelos SNNSIGMOID seleccionados en Hold-out.
- Archivo **snn_tl_offline.py**. Implementación de un modelo de Keras para SNNTLBASIC, SNNTLSEL y SNNTLC.
- Archivo **snn_tl_online.py**. Implementación del modelo de Keras para SNNTLONLINE.
- Archivo **snn_sigmoid.py**. Implementación del modelo de Keras para SNNSIGMOID.

La estructura del repositorio se resume a continuación:

```

1  entrenamiento_modelos_dl:
2      distance_functions.py
3      image_preprocessing_functions.py
4      macros.py
5      main_cv_snn_sigmoid.py
6      main_cv_snn_tlc.py
7      main_cv_snn_tl_basic.py
8      main_cv_snn_tl_online.py
9      main_cv_snn_tl_sel.py
10     main_hold_out_snn_sigmoid.py
11     main_hold_out_snn_tlc.py
12     main_hold_out_snn_tl_basic.py
13     main_hold_out_snn_tl_online.py
14     main_hold_out_snn_tl_sel.py
15     read_pkl_info.py
16     snn_sigmoid.py
17     snn_tl_offline.py
18     snn_tl_online.py
19     test_functions.py
20     threshold_detection_functions.py
21     utilities.py
22
23     generacion_imagenes_2d_craneales:
24         generar_imagenes.py

```

Las interdependencias entre los diferentes archivos se detallan en la Figura 4.5. Todos ellos, en conjunción, se utilizan para el entrenamiento, validación y test de los modelos de DL: el diagrama de cascada visualizado en la Figura 4.6 muestra el proceso seguido durante la ejecución de los archivos main.

4.3. Entorno de ejecución

El desarrollo de este TFG se ha llevado a cabo en dos diferentes entornos: por un lado, el diseño y pruebas iniciales de funcionamiento y, por otro,

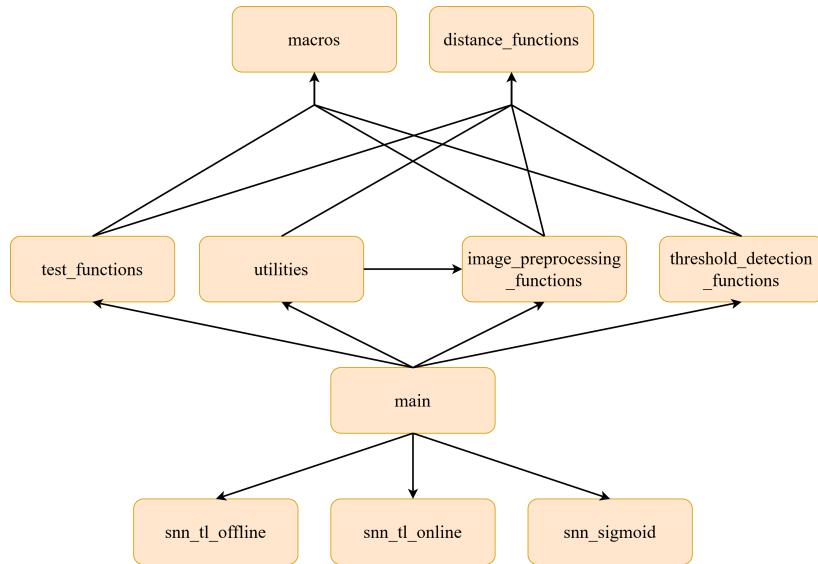


Figura 4.5: Interdependencias presentadas entre los diferentes archivos que componen el sistema diseñado e implementado.

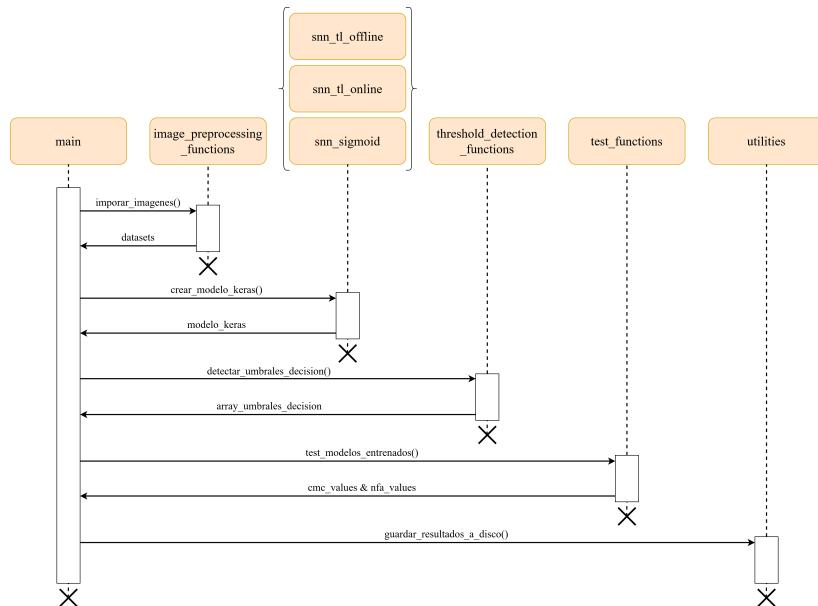


Figura 4.6: Diagrama de secuencia del sistema implementado, que muestra el proceso de interacción entre los diferentes archivos.

la ejecución de la experimentación diseñada. El lenguaje de programación utilizado es Python, bien mediante cuadernos tipo Jupyter Book, bien mediante scripts desarrollados en Spyder. Además, se ha utilizado TensorFlow [118] para el diseño y para la implementación de los modelos de DL.

Para el diseño y pruebas iniciales de funcionamiento se ha utilizado un ordenador con las siguientes especificaciones: HP Pavilion Laptop 14-bf1xx con un procesador Intel(R) Core(TM) i7-8550U CPU @ 1.8 GHz (bus a 2.00 GHz), 12Gb de memoria RAM y un dispositivo de almacenamiento SSD. Adicionalmente, las pruebas iniciales se realizaron en el entorno Google Colab [39], también conocido como Colaboratory; que permite programar y ejecutar Python desde un navegador de manera sencilla. No requiere configuración, permite compartir contenido fácilmente y, lo más importante, comparte de forma gratuita el acceso a GPUs.

Para la ejecución de la experimentación diseñada se utilizó la infraestructura disponible en el CPD Santa Lucía de la Universidad de Granada. Este centro de computación de altas prestaciones dispone de un acceso centralizado desde el cuál se admite la ejecución en las GPUs de los servidores instalados.

La infraestructura actual incluye los siguientes servidores:

- 1 servidor con 2 CPUs Intel(R) Xeon(R) E5-2630, 128 Gb RAM DDR4, 3 GPUs NVIDIA modelo GTX Titan X Pascal y 1 GPU NVIDIA modelo GTX Titan Xp.
- 1 servidor con 2 CPUs Intel(R) Xeon(R) E5-2630, 128 Gb RAM DDR4, 4 GPUs NVIDIA GTX Titan Xp.
- 1 servidor con 2 CPUs Intel(R) Xeon(R) E5-2630, 128 Gb RAM DDR3, 4 GPUs NVIDIA RTX 2080 Ti.
- 1 servidor con 2 CPUs Intel(R) Xeon(R) 4114, 128 Gb RAM DDR4, 4 GPUs NVIDIA Titan RTX.
- 2 servidores, cada uno con 2 CPUs Intel(R) Xeon(R) E5-2698, 512 Gb RAM DDR4, 8 GPUs NVIDIA Tesla V100 32Gb.
- 1 servidor con 2 CPUs Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz, 512 Gb RAM DDR4, 2 GPUs NVIDIA Quadro RTX 8000.

Toda la experimentación final se ha ejecutado en esta infraestructura debido al alto coste computacional de entrenar redes neuronales en el problema en cuestión.

Capítulo 5

Materiales y Métodos

5.1. Conjunto de datos

En este proyecto se trabaja sobre imágenes 2D de cráneos y de caras. Se ha dispuesto de dos *datasets* para ello: un conjunto de datos craneales e imágenes faciales cedido por varias instituciones y el *dataset* de imágenes faciales UTKFace [134], disponible públicamente.

5.1.1. Conjunto de datos craneales

En este proyecto se ha trabajado sobre un conjunto de modelos 3D craneales e imágenes faciales cedidos por las siguientes instituciones:

- Universidad de Granada, España
- Università degli Studi di Trieste, Italia
- Università degli Studi di Milano, Italia
- Vilnius University, Vilnius, Lituania
- University of Tennessee, Knoxville, EEUU

La Tabla 5.1 resume los puntos más importantes de los modelos 3D del conjunto de datos (el lector interesado puede consultar en la Tabla 7.1 del Apéndice A el detalle de cada modelo). El *dataset* está compuesto de 117 modelos 3D de cráneos, cada uno de una persona diferente (de ahora en adelante referidas como individuos, del PM1 al PM117), y 57 imágenes faciales de personas, asociadas a algunos de los anteriores individuos. No obstante, tres de los modelos 3D (PM24, PM40 y PM106) no se encuentran en condiciones correctas (ver Figura 5.1g) y, por tanto, no puede ser utilizados: se dispone solo de 114 modelos 3D craneales útiles. Cada una de las imágenes faciales disponibles corresponde a un individuo concreto, por lo que solo para la mitad de los individuos se cuenta tanto con un modelo 3D craneal como

con una imagen facial. A estos 57 individuos se les denominará *individuos positivos* (50 % de la muestra), mientras que a los 57 restantes, para los que únicamente se dispone del modelo 3D craneal, se les denominará *individuos negativos* (50 % restante de la muestra). La identificación del sexo de los individuos se ha realizado a través de las imágenes faciales, en los casos en los que están disponibles.

En cuanto a los modelos 3D craneales, hay 16 modelos que no disponen de mandíbula (ver Figura 5.1b), de los cuales en 12 de ellos el cráneo se encuentra, además, encima de un soporte cilíndrico vertical (ver Figura 5.1c). En 9 modelos el cráneo no presenta ni el hueso parietal ni el hueso occipital (ver Figura 5.1d), mientras que en 23 modelos aparece parcialmente el inicio de la columna vertebral (ver Figura 5.1e) y, en algunos casos, también parcialmente la clavícula (ver Figura 5.1f). En total hay 89 modelos en los que el cráneo aparece completo, de los cuales en 66 aparece únicamente el cráneo (ver Figura 5.1a). La diferencia de coloración de las imágenes es debida a que no se dispone de la textura para todos los modelos.

La mayoría de las mallas tiene entre 150 000 y 400 000 vértices y entre 300 000 y 800 000 caras, aunque hay 10 modelos que exceden el millón de vértices y caras.

| Característica | N _M | Característica | N |
|--|----------------|-------------------|------|
| Cráneos completos | 66 | Vértices (mínimo) | 62 |
| Sin los huesos parietal y occipital | 9 | Vértices (Q1) | 186 |
| Sin mandíbula | 4 | Vértices (Q3) | 430 |
| Sin mandíbula y sobre soporte | 12 | Vértices (máximo) | 2824 |
| Cráneo completo con parte de la columna vertebral | 23 | Caras (mínimo) | 119 |
| Modelos incorrectos | 3 | Caras (Q1) | 367 |
| Cráneos con imagen | 57 | Caras (Q3) | 852 |
| Cráneos sin imagen | 57 | Vértices (máximo) | 5018 |
| Hombres | 33 | | |
| Mujeres | 24 | | |
| Desconocidos | 57 | | |
| Total modelos correctos | 114 | | |

Tabla 5.1: Estadísticos de los modelos 3D craneales del conjunto de datos utilizado. En la izquierda, N_M es el número de modelos 3D que presentan la respectiva característica; en la derecha N es el número promedio en miles.

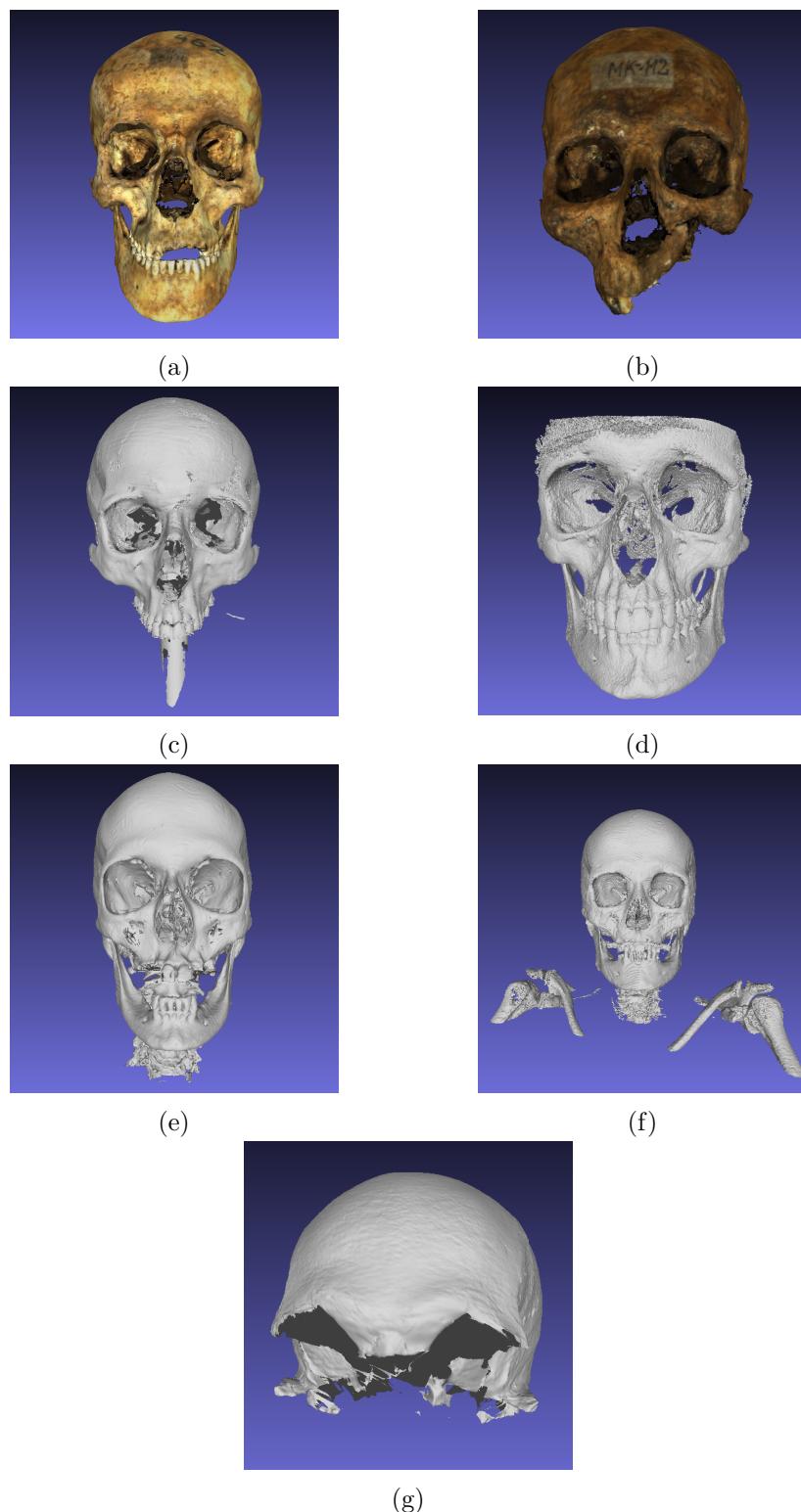


Figura 5.1: Diferentes ejemplos de los modelos craneales del conjunto de datos disponible: a) cráneo completo (PM9), b) cráneo sin mandíbula (PM87), c) cráneo sin mandíbula c/ soporte (PM3), d) cráneo sin parietal ni occipital (PM31), e) cráneo con presencia parcial de la columna vertebral (PM107), f) cráneo con presencia parcial de la columna vertebral y las clavículas (PM104), y g) modelo 3D inválido (PM40).

5.1.2. Conjunto de datos UTKFace

Esta *dataset* consta de más de 20.000 imágenes de caras de personas de diferentes edades, géneros y grupos étnicos. Las imágenes muestran variaciones en las expresiones faciales, las poses, la iluminación de la fotografía, la resolución, etc., componiendo un *dataset* con gran varianza.

El *dataset* es de acceso público, estando disponibles las imágenes originales y la versión recortada y alineada de las caras. Esta última opción ha sido la utilizada en este trabajo. En la Figura 5.2 se visualizan varios ejemplos de las imágenes de UTKFace.



Figura 5.2: Ejemplos de las imágenes faciales ya recortadas y alineadas del *dataset* UTKFace.

5.2. Métodos

Esta sección está enfocada a describir las diferentes metodologías utilizadas durante el desarrollo de la experimentación de este proyecto. En primer lugar, se detalla el procedimiento diseñado para el procesamiento del *dataset* de imágenes craneales. Luego se describe la CNN preentrenada utilizada, (*FaceNet*), se presentan las diferentes técnicas de DL utilizadas, tanto las tomadas de la literatura como las propuestas en este TFG, y se detalla el uso del umbral de decisión para la correspondencia de imágenes craneales y faciales. Finalmente, se recogen los diferentes modelos de DL propuestos en este TFG, junto con los hiperparámetros inherentes a cada uno de ellos.

5.2.1. Procesamiento del conjunto de datos craneales

En este trabajo se pretende explorar la utilización de imágenes 2D para establecer correspondencias entre cráneos y caras. El conjunto de datos craneales del que se dispone está compuesto de modelos 3D craneales, por lo que es necesario extraer imágenes 2D a partir de las mallas (el procedimiento seguido se describe en la siguiente Sección 5.2.1.3). Para ello, es preciso alinear todas las mallas entre sí y eliminar los elementos que añaden ruido a los modelos (por ejemplo, la columna vertebral), como se describe a continuación en las Secciones 5.2.1.1 y 5.2.1.2.

5.2.1.1. Alineamiento de los modelos 3D de los cráneos

El alineamiento de todos los modelos es necesario de cara a generar las imágenes de forma automatizada mediante un script de Python, manteniendo la misma orientación de los cráneos (frontal) en todos los casos. Para realizar el alineamiento de los modelos se ha utilizado la herramienta MeshLab [24]. Esta herramienta permite seleccionar tres puntos en cada par de mallas y extraer automáticamente la transformación a aplicar sobre la segunda para alinearla con la primera, como se puede ver en la Figura 5.3.

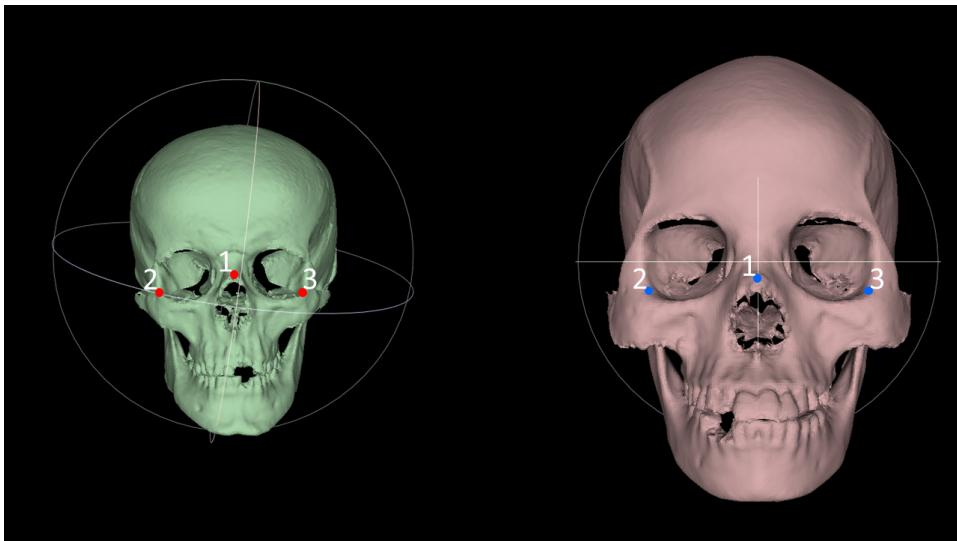


Figura 5.3: Selección de tres puntos en cada malla para extraer la transformación que alinee el modelo de la izquierda con el modelo de la derecha. La transformación hallada transportará los puntos 1, 2 y 3 (puntos en rojo) del modelo de la izquierda a los puntos 1, 2 y 3 (puntos en azul) del modelo de la derecha. Elaboración propia.

5.2.1.2. Eliminación de elementos no deseables en los modelos 3D

Para mejorar la calidad de los modelos disponibles, se han eliminado de estos aquellas partes innecesarias, como los soportes verticales o las columnas vertebrales (ver Figura 5.4). Para ello, se han seleccionado y eliminado los vértices que conforman estos elementos en los modelos 3D.

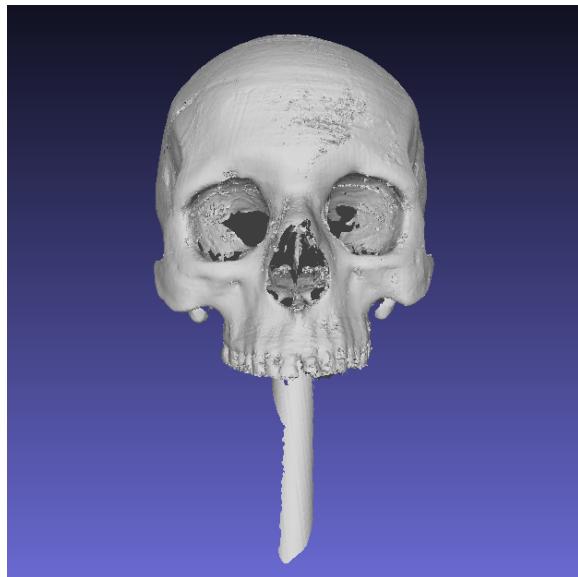


Figura 5.4: Algunos modelos 3D del *dataset* presentan imperfecciones: soportes verticales, fragmentos de la columna vertebral, etc. En la imagen se ve el modelo 3D del individuo PM73, que presenta un soporte vertical debajo del cráneo.

5.2.1.3. Generación de imágenes a partir de los modelos 3D

El objetivo de este proyecto es explorar la ID a partir de imágenes craneales y faciales. Como el *dataset* del que se dispone contiene modelos 3D craneales, ha sido necesario definir un método de obtención de imágenes a partir de estos modelos 3D de tal forma que las imágenes craneales resultantes sean imágenes frontales del cráneo. Se ha utilizado la librería VTK [104] mediante su implementación en Python para la visualización de las mallas de los modelos 3D craneales y para generar las imágenes a partir de estas.

Las herramientas de VTK permiten generar escenas 3D en la que se visualicen los modelos 3D craneales y modificar la malla (por ejemplo, rotándola respecto a un eje) y la cámara que visualiza la escena. Para la visualización de los cráneos se ha obtenido el centro de estos y se ha situado la cámara en la recta paralela al eje Z que pasa por dicho centro, siendo el centro el

punto medio de los extremos en cada eje:

$$\text{centro} = \left(\frac{\min_x + \max_x}{2}, \frac{\min_y + \max_y}{2}, \frac{\min_z + \max_z}{2} \right) \quad (5.1)$$

Donde \min_x , \min_y y \min_z son las coordenadas mínimas de la malla en cada eje y \max_x , \max_y y \max_z son las coordenadas máximas de la malla en cada eje. En la Figura 5.5 se visualiza el centro de un cráneo.

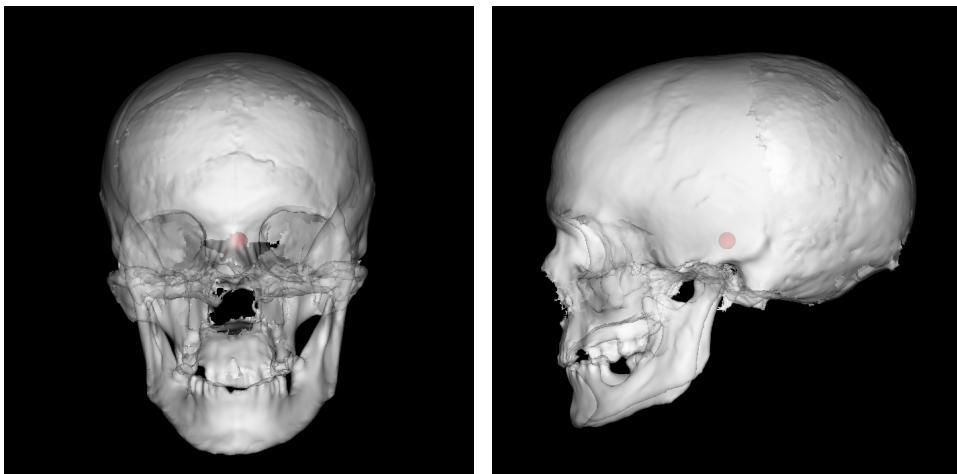


Figura 5.5: El centro de un cráneo (en rojo) es el punto medio de los extremos del modelo 3D en cada eje. Elaboración propia.

El procedimiento seguido para generar imágenes a partir de cada modelo 3D craneal es el descrito en Algoritmo de generación aleatoria de imágenes. Este algoritmo introduce cierta aleatoriedad en las imágenes 2D, replicando un escenario real en el que la perspectiva de los cráneos puede no ser la misma siempre. Hay dos puntos importantes a tener en cuenta en la implementación del anterior método:

1. Ajuste del ángulo de visión de la cámara tal que el cráneo llene aproximadamente la ventana de visualización. Depende de la distancia a la que se posiciona la cámara respecto al cráneo.
2. Máximo desplazamiento que se puede aplicar sobre el cráneo tal que este se continúe visualizando completamente dentro de la ventana. Depende del anterior ángulo de visión y de la distancia a la que se posiciona la cámara respecto al cráneo.

Al posicionar la cámara en un punto C a una distancia concreta del cráneo, para cada vértice P del cráneo existen dos ángulos de visión mínimos α_1 y α_2 tales que P se visualiza en el borde horizontal o vertical de la ventana

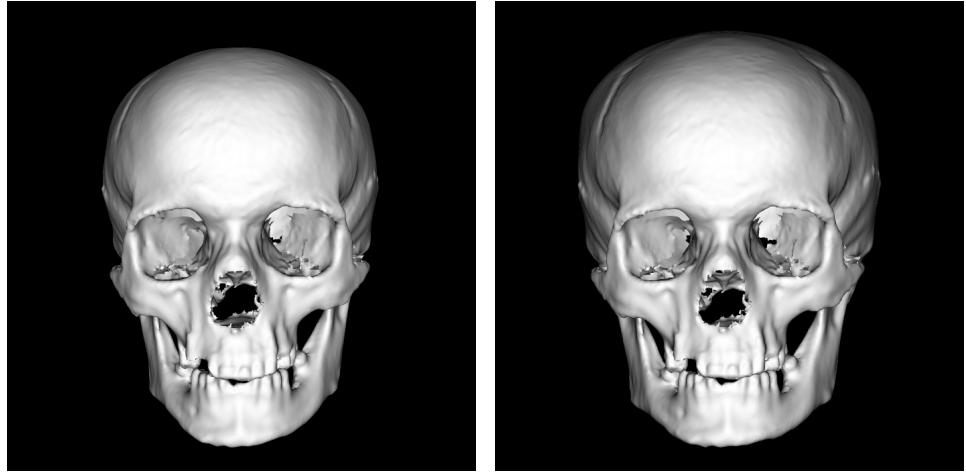


Figura 5.6: Diferentes perspectivas de un cráneo en función de distancia de la cámara al mismo. A la izquierda, perspectiva de un cráneo con la cámara situada a 0.6 metros del mismo. A la derecha, perspectiva del mismo un cráneo con la cámara situada a 3 metros del éste. Elaboración propia.

de visualización. Como se puede ver en la Figura 5.7, α_1 y α_2 dependen de la posición relativa del punto a la cámara:

$$\alpha_1 = \arctan \left(\frac{\text{abs}(P_x - C_x)}{\text{abs}(C_z - P_z)} \right) \quad (5.2)$$

$$\alpha_2 = \arctan \left(\frac{\text{abs}(P_y - C_y)}{\text{abs}(C_z - P_z)} \right) \quad (5.3)$$

El máximo de estos dos ángulos, $\alpha = \max(\alpha_1, \alpha_2)$, es el mínimo ángulo de visión de la cámara que permite visualizar P dentro de la ventana. Para el cráneo en conjunto, existe un ángulo de visión mínimo, anguloVision (ver Ecuación 5.4), tal que la malla entera se visualiza en la ventana y es obtenido como el máximo de los ángulos de visión mínimos, α , de cada uno de los vértices que componen el modelo 3D.

$$\text{anguloVision} = \sum_{i=1}^N \alpha_i \quad (5.4)$$

Donde α_i es el ángulo de visión mínimo del vértice i de la malla.

Una vez fijado el ángulo de visión, α , y la posición, C , de la cámara, antes de aplicar un desplazamiento sobre el cráneo se debe comprobar que dicho desplazamiento no elimine parte del modelo (porque salga fuera de la ventana de visualización). Para ello, es necesario calcular el máximo desplazamiento en los ejes X e Y y en cada sentido (máximo desplazamiento positivo y máximo desplazamiento negativo).

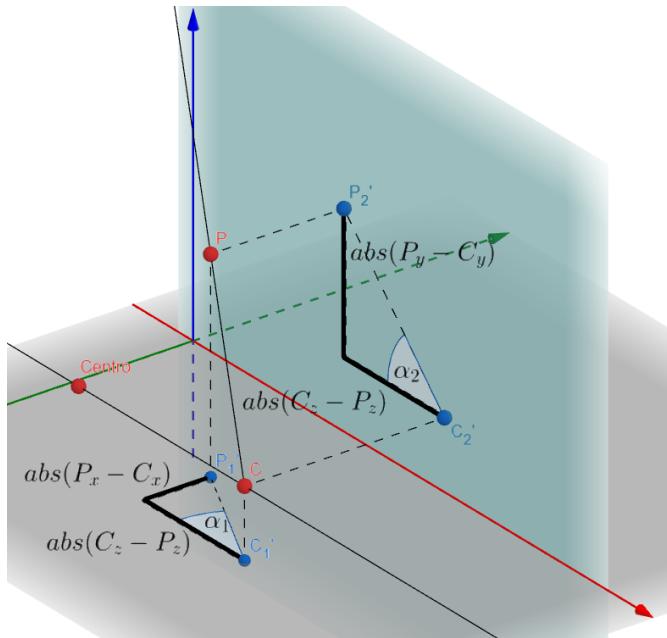


Figura 5.7: Gráfica que ilustra la obtención de los ángulos de visión mínimos, α_1 (horizontal) y α_2 (vertical), a partir de la posición de la cámara, situada en el punto C , y del punto P del cráneo. C'_1 y P'_1 son las proyecciones de estos dos puntos en el plano XZ y C'_2 y P'_2 son las proyecciones en el plano YZ. Notar que el punto *Centro* se corresponde con el centro del cráneo, calculado según la Ecuación 5.1, y que la cámara se posiciona sobre una recta paralela al eje Z que pasa por este punto. Elaboración propia.

Un punto P puede desplazarse positivamente y negativamente en los ejes X e Y: su máximo desplazamiento positivo, $desp_{posx}$, y negativo, $desp_{negx}$, en el eje X y su máximo desplazamiento positivo, $desp_{posy}$ y negativo, $desp_{negy}$, en el eje Y se calculan como (Ecuaciones 5.5, 5.6, 5.7 y 5.8):

$$desp_{posx}(P) = abs(abs(C_Z - P_Z) \tan \alpha - (P_X - C_X)) \quad (5.5)$$

$$desp_{negx}(P) = -abs(-abs(C_Z - P_Z) \tan \alpha - (P_X - C_X)) \quad (5.6)$$

$$desp_{posy}(P) = abs(abs(C_Z - P_Z) \tan \alpha - (P_Y - C_Y)) \quad (5.7)$$

$$desp_{negy}(P) = -abs(-abs(C_Z - P_Z) \tan \alpha - (P_Y - C_Y)) \quad (5.8)$$

Los máximos desplazamientos del modelo 3D completo dependen de los $desp_{posx}$, $desp_{negx}$, $desp_{posy}$ y $desp_{negy}$ de sus vértices de la siguiente forma:

- El máximo desplazamiento positivo en el eje X, $\max_{desp_{posx}}$ (Ecuación 5.9), es el mínimo $desp_{posx}$ de los puntos PD cuya coordenada X es mayor que la coordenada X de C (puntos que están a la derecha de la cámara).
- El máximo desplazamiento negativo en el eje X, $\max_{desp_{negx}}$ (Ecuación 5.10), es el máximo $desp_{negx}$ de los puntos PI cuya coordenada X es menor que la coordenada X de C (puntos que están a la izquierda de la cámara). En este caso se trata del máximo debido a que los $desp_{negx}$ son negativos; se corresponde con el mínimo $abs(desp_{negx})$.
- El máximo desplazamiento positivo en el eje Y, $\max_{desp_{posy}}$ (Ecuación 5.11), es el mínimo $desp_{posy}$ de los puntos PA cuya coordenada Y es mayor que la coordenada Y de C (puntos que están arriba de la cámara).
- El máximo desplazamiento negativo en el eje Y, $\max_{desp_{negy}}$ (Ecuación 5.12), es el máximo $desp_{negy}$ de los puntos PD cuya coordenada Y es menor que la coordenada Y de C (puntos que están abajo de la cámara). En este caso se trata del máximo debido a que los $desp_{negy}$ son negativos; se corresponde con el mínimo $abs(desp_{negy})$.

$$\max_{desp_{posx}} = \min(desp_{posx}(PD_i)) \quad (5.9)$$

$$\forall PD_i \in PD$$

$$\max_{desp_{negx}} = \max(desp_{negx}(PI_i)) \quad (5.10)$$

$$\forall PI_i \in PI$$

$$\max_{desp_{posy}} = \min(desp_{posy}(PA_i)) \quad (5.11)$$

$$\forall PA_i \in PA$$

$$\max_{desp_{negy}} = \max(desp_{negy}(PD_i)) \quad (5.12)$$

$$\forall PD_i \in PD$$

Algoritmo de generación aleatoria de imágenes

1. Recibir el modelo 3D craneal.
2. Crear una ventana de visualización con VTK y mostrar el cráneo en ella.

3. Realizar una rotación aleatoria del cráneo en los tres ejes. Se genera una rotación aleatoria en cada eje (cada una con diferente ángulo), todas en el rango $[-10^\circ, 10^\circ]$.
4. Posicionamiento de la cámara a 0.6 metros o a 3 metros del bounding box del cráneo. La cámara está situada sobre una recta paralela al eje Z en el centro del cráneo, por lo que modificar la distancia de la cámara al cráneo equivale a modificar únicamente la coordenada Z de la cámara. En la Figura 5.6 se puede comprobar la diferencia que se aprecia en la perspectiva del cráneo dependiendo de la distancia a la que está situada la cámara: cuanto más cerca se coloque la cámara mayor es la diferencia de tamaño entre los elementos del cráneo más cercanos y los más lejanos. Además, al posicionar la cámara a 0.6 metros hay elementos de la de los huesos parietal y temporal del cráneo que dejan de ser visibles.
5. Ajuste del ángulo de visión con la Ecuación 5.4.
6. Computar los máximos desplazamientos aleatorios positivos y negativos en los ejes X e Y, $max_{desp_{posx}}$, $max_{desp_{negx}}$, $max_{desp_{posy}}$ y $max_{desp_{negy}}$.
7. Realizar un desplazamiento aleatorio en el eje X entre $max_{desp_{negx}}$ y $max_{desp_{posx}}$.
8. Realizar un desplazamiento aleatorio en el eje Y entre $max_{desp_{negy}}$ y $max_{desp_{posy}}$.
9. Tomar una captura de la ventana de visualización, generando una imagen 2D del modelo 3D de entrada al algoritmo.

5.2.2. FaceNet

FaceNet es una CNN diseñada para el reconocimiento facial y que presenta una arquitectura prácticamente idéntica a la de *GoogleLeNet* [105]. La red contiene una combinación de diferentes tipos de capas (convolucionales, densas, pooling, etc.) con un total de 7.5 millones de parámetros.

El principio básico de la estructura de *FaceNet* son los módulos *Inception*, pequeños bloques de capas que combinan convoluciones de diferentes tamaños [113]. La idea detrás de estos módulos es la extracción de características a diferentes niveles, ya que incluyen capas convolucionales con filtros de tamaño 1×1 , 3×3 y 5×5 , como se puede ver en la Figura 5.9. Además, para reducir el coste computacional de la red, los módulos Inception con reducción de dimensionalidad aplican una convolución 1×1 antes de las convoluciones con filtros de mayor tamaño y después del *max pooling*. Esto permite reducir la dimensionalidad del volumen de entrada, haciendo

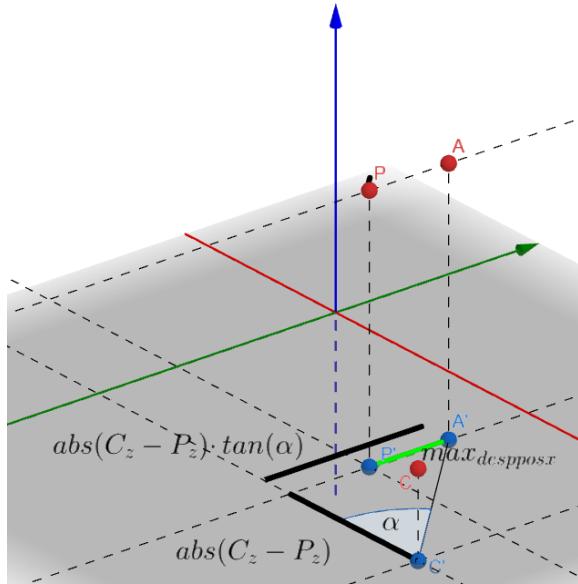


Figura 5.8: Gráfica que ilustra la obtención del máximo desplazamiento positivo aplicable en el eje X, $max_{despposx}$ en verde, sobre un punto P del cráneo dado un ángulo de visión α fijo y la cámara posicionada en el punto C . El punto A es el punto más alejado que se visualiza en la recta paralela al eje X por el punto P , siendo el punto más alejado al que se puede trasladar el punto P sin dejar de visualizarlo. C' , P' y A' son las proyecciones de, respectivamente, los puntos C , P y A en el plano XZ. Elaboración propia.

que el número de convoluciones de filtros 3×3 y 5×5 se vea reducido y que el coste computacional de la red disminuye notablemente (aunque haya que realizar convoluciones 1×1 previas, estas son menos costosas).

La arquitectura de *FaceNet* se puede ver en la Tabla 5.2: una característica clave de esta red es que fue diseñada para producir *embeddings* 128-dimensionales y ser entrenada como SNN, para lo cual utiliza elementos como capas de pooling de norma L2.

5.2.3. Modelos implementados para realizar la identificación craneofacial a partir de fotografías de cráneos y caras

El objetivo de este TFG es utilizar un modelo de DL que reciba imágenes craneales y faciales y permita realizar directamente identificaciones. Se pretende proponer un modelo que, recibiendo un imagen craneal, prediga la imagen facial correspondiente; en otras palabras, un modelo que identifique que cara está vinculada al cráneo. El escenario real consiste en comparar la imagen craneal con una serie de imágenes faciales contenidas en una base de datos (BD), estableciendo la identificación con la más semejante

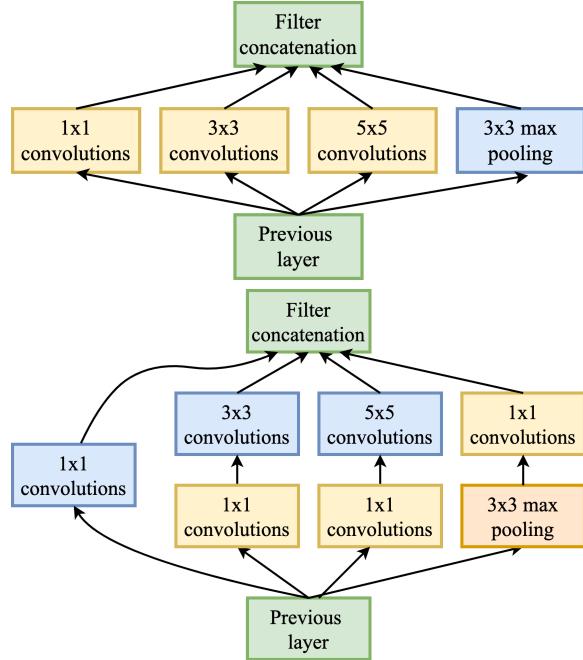


Figura 5.9: Estructura de los módulos Inception: en la parte superior la versión Naive, en la inferior la versión con reducción de dimensionalidad. Elaboración propia a partir de las imágenes mostradas en [113].

| Tipo de capa | output size | depth | #1x1 | #3x3r | #3x3 | #5x5r | #5x5 | pool proj | n p | FLOPS |
|------------------|-------------|-------|------|-------|-------|-------|-------|-----------|------|-------|
| conv1 (7x7x3,2) | 112x112x64 | 1 | | | | | | | 9K | 119M |
| max pool + norm | 56x56x64 | 0 | | | | | | 3 × 3, 2 | | |
| inception (2) | 56x56x192 | 2 | | 64 | 192 | | | | 115K | 360M |
| norm + max pool | 28x28x192 | 0 | | | | | | 3 × 3, 2 | | |
| inception (3a) | 28x28x256 | 2 | 64 | 96 | 128 | 16 | 32 | 32p | 164K | 128M |
| inception (3b) | 28x28x320 | 2 | 64 | 96 | 128 | 32 | 64 | L2, 64p | 228K | 179M |
| inception (3c) | 14x14x640 | 2 | 0 | 128 | 256,2 | 32 | 64,2 | 3 × 3, 2 | 398K | 108M |
| inception (4a) | 14x14x640 | 2 | 256 | 96 | 192 | 32 | 64 | L2, 128p | 545K | 107M |
| inception (4b) | 14x14x640 | 2 | 224 | 112 | 224 | 32 | 64 | L2, 128p | 595K | 117M |
| inception (4c) | 14x14x640 | 2 | 192 | 128 | 256 | 32 | 64 | L2, 128p | 654K | 128M |
| inception (4d) | 14x14x640 | 2 | 160 | 144 | 288 | 32 | 64 | L2, 128p | 722K | 142M |
| inception (4e) | 7x7x1024 | 2 | 0 | 160 | 256,2 | 64 | 128,2 | 3 × 3, 2 | 717K | 56M |
| inception (5a) | 7x7x1024 | 2 | 384 | 192 | 384 | 48 | 128 | L2, 128p | 1.6M | 78M |
| inception (5b) | 7x7x1024 | 2 | 384 | 192 | 384 | 48 | 128 | m, 128p | 1.6M | 78M |
| lavg pool | 1x1x1024 | 0 | | | | | | | | |
| fully conn | 1x1x128 | 1 | | | | | | | 131K | 0.1M |
| L2 normalization | 1x1x128 | 0 | | | | | | | | |
| total | | | | | | | | | 7.5M | 1.6B |

Tabla 5.2: Arquitectura de *FaceNet*. *output size* es la dimensionalidad del volumen de salida, *depth* es el número de capas contenidas en cada bloque, *#1x1*, *#3x3* y *#5x5* son el número de filtros de cada tamaño, *#3x3r* y *#5x5r* son el número de filtros 1×1 antes de las convoluciones, *pool proj* es el número de filtros 1×1 después del pooling y *n p* y *FLOPS* es el número de parámetros y operaciones matemáticas asociadas a cada bloque.

Debido a la escasez de estudios en este problema y la ausencia de estudios revisados que empleen técnicas de DL, se han diseñado varios modelos de DL para poder hacer una comparativa entre su rendimiento. En las siguientes secciones se detallan los diferentes modelos.

5.2.3.1. Red Neuronal Siamesa con Triplet Loss

Se ha diseñado una arquitectura de SNN pensada para ser entrenada con TL. Esta SNN recibe como entrada triplets de ejemplos, donde el ancla es una imagen facial, el ejemplo positivo (de ahora en adelante *positivo*) es una imagen craneal de la misma persona y el ejemplo negativo (de ahora en adelante *negativo*) es una imagen craneal de otra persona diferente. El generador de *embeddings* de la SNN es una CNN (replicada tres veces, con pesos compartidos), que genera como salida los *embeddings* de cada una de las imágenes que componen las triplets de entrada de la red.

Un aspecto importante a remarcar en el empleo de las SNN entrenadas con TL es que su funcionamiento durante el entrenamiento es diferente a su funcionamiento durante test, es decir, a la hora de realizar nuevas predicciones. Por un lado, el entrenamiento de este tipo de SNNs permite aprender los pesos del generador de *embeddings* subyacente a partir de los triplets de imágenes de entrada. Por otro lado, la predicción de un par de imágenes (determinar si una imagen craneal es de la misma persona que una imagen facial) es realizada empleando únicamente el generador de embeddings. Ello se realizará generando el *embedding* de la imagen craneal y el *embedding* de la imagen facial y comparándolos entre sí mediante una función de distancia. La Figura 5.10 ilustra el proceso completo de predicción de una imagen craneal con una BD de imágenes faciales.

La red es entrenada utilizando la función de pérdida TL (Ecuación 5.13), cuyo objetivo es conseguir que las distancias entre *embeddings* de imágenes del mismo individuo sea menor que la distancia entre *embeddings* de imágenes de individuos diferentes. Para una tripla de entrenamiento (A, P, N) el modelo intentará aprender a disminuir $d(A, P)$ y aumentar $d(A, N)$. Esto se traduce en que los *embeddings* de pares cara-cráneo del mismo individuo estarán a menor distancia que los *embeddings* de pares donde el cráneo y la cara pertenecen a dos individuos diferentes.

$$TL(A, P, N) = \max\left(0, d(E(A), E(P)) - d(E(A), E(N)) + \alpha\right) \quad (5.13)$$

Un aspecto clave en el entrenamiento de las SNN con TL es la selección de triplets para el entrenamiento [105]. Esta función de pérdida incentiva que el modelo separe la distancia entre *embeddings* ancla-positivo fuera de un margen α respecto a la distancia entre *embeddings* ancla-negativo. Para simplificar, de ahora en adelante se va a mencionar que dos imágenes están cercanas referenciando al hecho de que su *embeddings* están cercanos, es

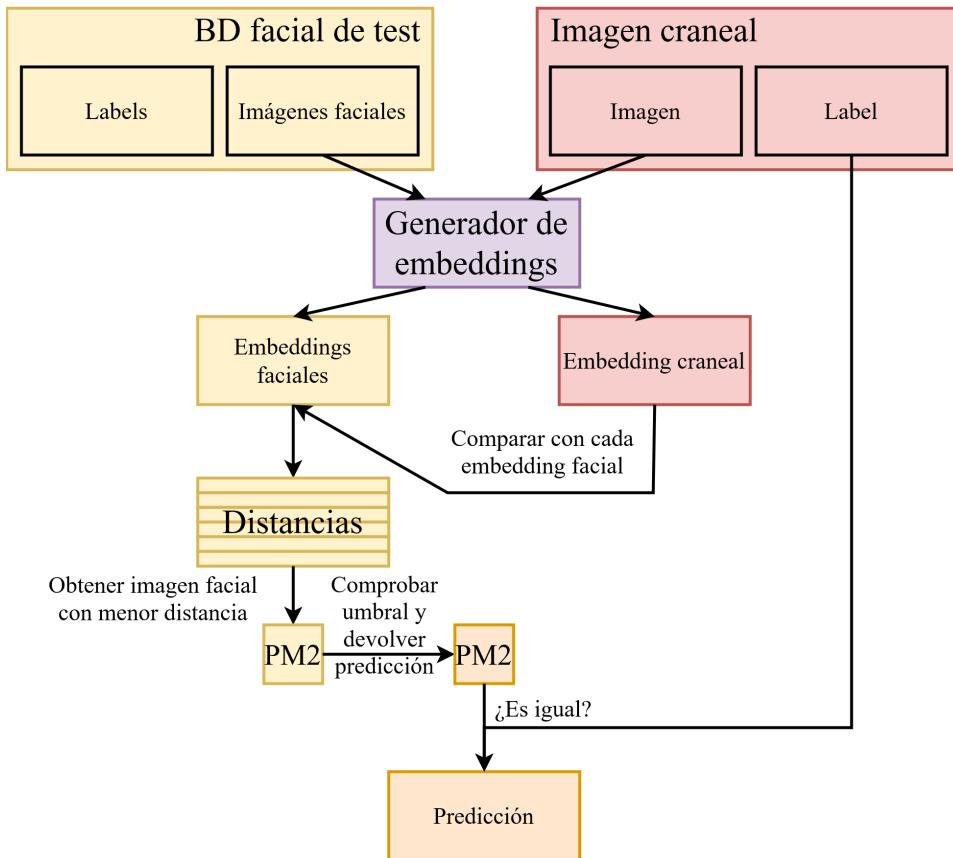


Figura 5.10: Proceso de predicción de las imágenes craneales en las SNN entrenadas con TL. Inicialmente se generan los *embeddings* de la BD facial y de la imagen craneal. Tras ello, cada *embedding* facial es comparado con el *embedding* craneal, obteniendo una distancia para cada una de las comparaciones. La imagen facial que menor distancia obtiene es la predicha, siempre que esta distancia sea menor al umbral de decisión (ver Sección 5.2.3.3).

decir, que la distancia entre ellos es pequeña. En el caso de las tripletas de entrenamiento, hay tres escenarios posibles (la Figura 5.11 refleja los tres escenarios):

- **Tripletas sencillas.** Son aquellas cuya pérdida es 0, debido a que se cumple que $d(E(A), E(P)) + \alpha < d(E(A), E(N))$, donde $E(x)$ es el *embedding* de x y d es la función de distancia. Estas tripletas no proporcionan ningún aprendizaje a la red, ya que la función de pérdida es 0 (en este caso el optimizador no actualiza los pesos de la red).
- **Tripletas complicadas o *hard*.** Son aquellas en las que la imagen negativa está más cercana al ancla que la positiva: $d(E(A), E(N)) <$

$d(E(A), E(P))$. Estas tripletas son las que mayor conocimiento proporcionan a la red durante el entrenamiento de esta, ya que se ve forzada a adaptar sus pesos para aumentar la distancias ancla-negativo y disminuir la distancia ancla-positivo.

- Tripletas semi-complicadas o *semi-hard*. Son aquellas en las que la imagen negativa no está más cercana al ancla que la positiva, pero aún así tienen función de pérdida positiva. Estas tripletas cumplen la condición $d(E(A), E(P)) < d(E(A)$ y la condición $d(E(N)) < d(E(A), E(P)) + \alpha$: la negativa no es más cercana que la positiva pero todavía está dentro del margen α que caracteriza las distancias ancla-positivo.

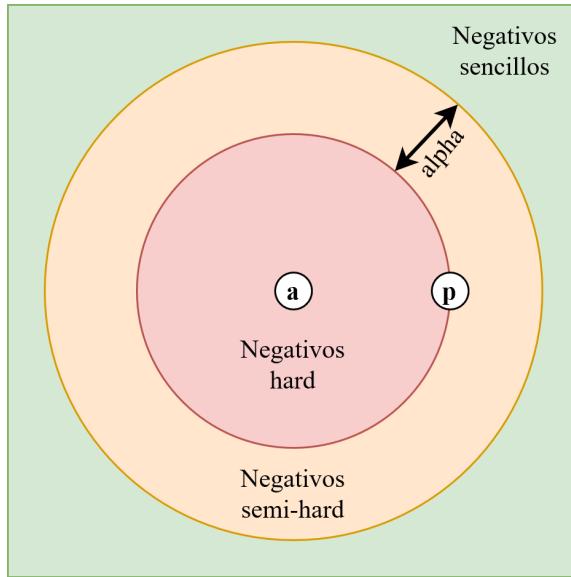


Figura 5.11: Hay tres escenarios posibles en las tripletas de entrenamiento: *negativos sencillos* (donde el negativo está fuera del margen de distancias del positivo), *negativos semi-hard* (donde el negativo está más alejado que el positivo pero todavía dentro del margen) y *negativos hard* (donde el negativo está más cerca que el positivo). Elaboración propia a partir de la imagen mostrada en [83].

Durante el entrenamiento de una red, cuanto mayor sea el número de tripletas sencillas, menor es el aprendizaje conseguido [47]. En otras palabras, si se desea conseguir un alto grado de aprendizaje es necesario proporcionar únicamente tripletas *hard* y tripletas *semi-hard* a la red, desechar las tripletas sencillas porque perjudican el entrenamiento. Por esta razón, el método de generación de tripletas de entrenamiento resulta fundamental en el rendimiento de las SNN.

Se pueden definir dos categorías en los métodos de generación de tripletas: **generación offline** y **generación online**. La primera de ellas consiste

en generar las tripletas de entrenamiento antes de comenzar este, mientras que en la segunda categoría las tripletas son seleccionadas dinámicamente a partir de las de imágenes de entrenamiento (recibidas por separado, sin formar tripletas predefinidas). Las estrategias de generación online son más costosas computacionalmente (teniendo en cuenta únicamente el aspecto de producir las tripletas), ya que en cada iteración del entrenamiento se deben producir nuevas tripletas: en la generación offline todas las tripletas son generadas en un único paso previo al entrenamiento. Un modelo con generación offline es alimentado (durante el entrenamiento) con tripletas ya definidas; un modelo con generación online es alimentado con imágenes aisladas que son combinadas durante el entrenamiento para formar las tripletas.

En este trabajo se han diseñado y adaptado tres estrategias de generación de tripletas, una de ellas utilizando generación offline y las dos restantes utilizando generación online. En el caso de las estrategias de generación online, el modelo es alimentado con *batches* de tres secuencias de imágenes (anclas, positivos y negativos) con sus respectivas etiquetas y las tripletas se generan en cada iteración de un *batch*. Contrariamente, en la estrategia de generación offline el modelo es alimentado por las tripletas sin sus etiquetas (ya que no son necesarias al ya estar formados los ejemplos de entrenamiento). Las tres estrategias se detallan a continuación, junto con el método de cálculo de la TL en cada *batch*.

Estrategia de generación offline de tripletas

Se trata de una estrategia de generación aleatoria: se crea una tripla para cada par ancla-positivo disponible y se completa escogiendo aleatoriamente un ejemplo negativo. No se generan todas las tripletas posibles: no se combinan todos los pares ancla-positivo disponibles con todos los ejemplos negativos, ya que esto provocaría que el entrenamiento fuese inviable computacionalmente.

Debido a la cierta probabilidad de existencia de tripletas sencillas y como estas no aportan conocimiento a la red, el rendimiento de la red neuronal se puede ver mermado. De cara a solucionar este problema, en este TFG se implementan dos soluciones: una propuesta de selección online de tripletas *semi-hard* y *hard* tras su generación offline aleatoriamente y la modificación de la TL denominada *Triplet Loss Condicional* (TLC) [108].

La primera consiste en seleccionar, durante el propio entrenamiento de la red, únicamente las tripletas *semi-hard* y *hard* (ver Figura 5.12) para el cómputo de la función de pérdida.

La segunda, que no es una propuesta específica de este TFG, trata de disminuir el aspecto negativo de la generación aleatoria de tripletas mediante la introducción de términos de penalización o recompensa en función de la naturaleza de las tripletas. Se establece una nueva clasificación de tripletas: mejores tripletas TLC (aquellas que cumplen la Ecuación 5.14) y peores

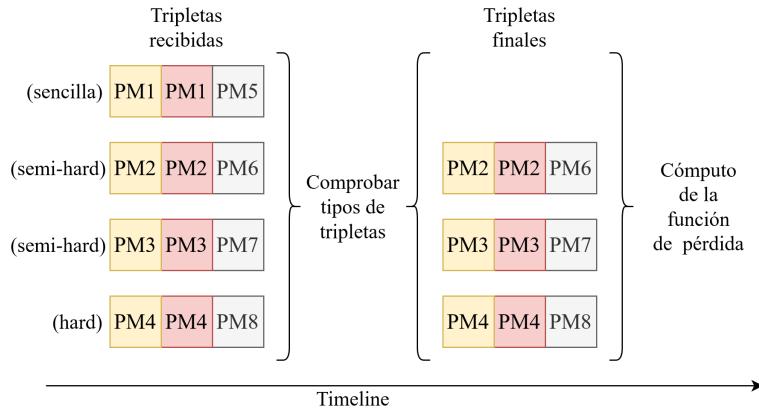


Figura 5.12: La selección online de tripletas consiste en eliminar aquellas tripletas sencillas y solo computar la función de pérdida sobre las tripletas *semi-hard* y *hard*. Para ello es necesario comprobar el tipo de las tripletas antes de calcular la función de pérdida.

tripletas TLC (las que satisfacen la Ecuación 5.15).

$$\begin{aligned} \epsilon &< d(E(A), E(P)) - d(E(A), E(N)) + \alpha \leq 2\epsilon, \\ \epsilon &= k\alpha, \quad 0 < k < 1 \end{aligned} \quad (5.14)$$

$$d(E(A), E(N)) + \alpha < d(E(A), E(P)) \quad (5.15)$$

La TLC añade una recompensa (Ecuación 5.16) a la función de pérdida para las mejores tripletas TLC y una penalización (Ecuación 5.17) para las peores tripletas TLC, permitiendo así que la red tenga más en cuenta aquellas tripletas equívocamente clasificadas.

$$TLC_r(A, P, N) = TL(A, P, N) - \alpha_{pen} \frac{d(E(A), E(P)) + d(E(A), E(N))}{2} \quad (5.16)$$

$$TLC_p(A, P, N) = TL(A, P, N) + \alpha_{pen} \frac{d(E(A), E(P)) + d(E(A), E(N))}{2} \quad (5.17)$$

Donde α_{pen} regula la intensidad de la recompensa y la penalización. La función de pérdida TLC para una tripleta de entrenamiento se define de la siguiente manera:

$$TLC(A, P, N) = \begin{cases} TLC_r(A, P, N) & \text{si cumple la Ecuación 5.14} \\ TLC_p(A, P, N) & \text{si cumple la Ecuación 5.15} \\ TL(A, P, N) & \text{en otro caso} \end{cases} \quad (5.18)$$

Por otro lado, se han restringidos los *embeddings* limitando su norma ($\|E(x)\|^2 = 1$) mediante un término de regularización (Ecuación 5.19, donde λ regula la intensidad de la regularización) añadido a las funciones de pérdida TL y TLC. Con esta penalización se pretenden dos objetivos: conseguir una mejor generalización de la red y evitar que la red incremente los coeficientes de los *embeddings* para satisfacer la restricción de la TL fácilmente.

$$\lambda(\|E(A)\|^2 + \|E(P)\|^2 + \|E(N)\|^2) \quad (5.19)$$

Estrategia de generación online de tripletas batch-all

Mediante esta estrategia se generan todas las posibles tripletas *semi-hard* y *hard* a partir de las imágenes recibidas en el *batch*. Esto implica que cada imagen facial es unida a todas las imágenes craneales del mismo individuos y cada uno de estos pares ancla-positivo es unido a todas las imágenes craneales negativas del batch, descartando aquellas tripletas sencillas que se hayan generado (ver Figura 5.13). Al generar dinámicamente todas las posibles tripletas y poder descartar aquellas sencillas se consigue un mayor volumen de tripletas *semi-hard* y *hard* (respecto a la generación offline), que proporcionan mayor conocimiento a la red. La función de pérdida es únicamente computada en las tripletas *semi-hard* y *hard* y no en las tripletas sencillas, por lo que estas últimas no ralentizan el entrenamiento y disminuyen el rendimiento de la red.

El procedimiento completo de generación online de tripletas *batch-all* se puede consultar en la Sección Estrategia de generación online de tripletas batch-all del Apéndice 7.

Estrategia de generación online batch-hard de tripletas

Este método genera, por cada imagen facial (ancla) recibida en el *batch*, una única triplete: la más complicada disponible en el *batch*, para lo cual se busca la imagen craneal de la misma persona a mayor distancia y la imagen craneal de otra persona a menor distancia (ver Figura 5.14). Las tripletas generadas son las que mayor función de pérdida tienen en el *batch* para cada una de las imágenes faciales, por lo que se consiguen las tripletas que mayor información le pueden proporcionar a la red.

El procedimiento completo de generación online de tripletas *batch-hard* se puede consultar en la Sección Estrategia de generación online de tripletas batch-hard del Apéndice 7.

5.2.3.2. Red Neuronal Siamesa con salida sigmoidal

En este TFG se propone una arquitectura de SNN con salida sigmoidal, que recibe dos imágenes (una facial y otra craneal) y devuelve un valor en el rango $[0, 1]$ que mide la similitud entre ambas. Esta red está compuesta

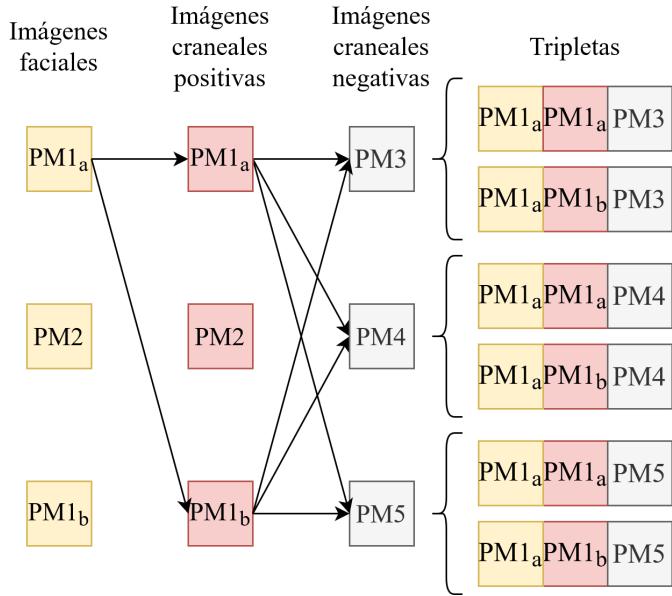


Figura 5.13: La estrategia generación online de tripletas *batch-all* consiste en emparejar cada imagen facial con todas las imágenes craneales positivas del mismo individuo y todos estos pares, a su vez, con todas las imágenes craneales negativas del *batch*. En el ejemplo solo se visualizan las tripletas generadas para la imagen facial PM1_a (en amarillo), perteneciente al individuo PM1: esta es emparejada con las imágenes craneales PM1_a y PM1_b (en rojo) y con todas las imágenes craneales negativas (en gris, PM3, PM4 y PM5).

de un generador de *embeddings* (una CNN) replicado dos veces, creando dos sub redes cuya salida es concatenada y alimentada a un modelo que genera la salida sigmoidal final (ver Figuras 5.15 y 5.16). La red recibe pares de imágenes cara-cráneo, que pueden ser positivos (si ambas imágenes pertenecen al mismo individuo) o negativos (si las imágenes pertenecen a individuos diferentes). La predicción correcta para los pares positivos sería 1, mientras que la correcta para los negativos sería 0.

Se ha decidido utilizar la función de pérdida *MSE* (Ecuación 5.20) para el entrenamiento de este tipo de red. Esta función mide el grado de diferencia de las etiquetas reales y y las etiquetas predichas \hat{y} de los datos de entrenamiento. La MSE promedia el error cometido en las predicciones por el modelo de tal forma que los errores grandes son más penalizados (por la función cuadrática de la diferencia de etiquetas). La motivación del uso de esta función viene dada por el interés en la tarea de IDH (más allá de realizar identificaciones positivas) de reducir el número de posibles candidatos. En este sentido, puede ser tolerable cometer errores pequeños tales que el modelo no consiga realizar identificaciones positivas con gran cer-

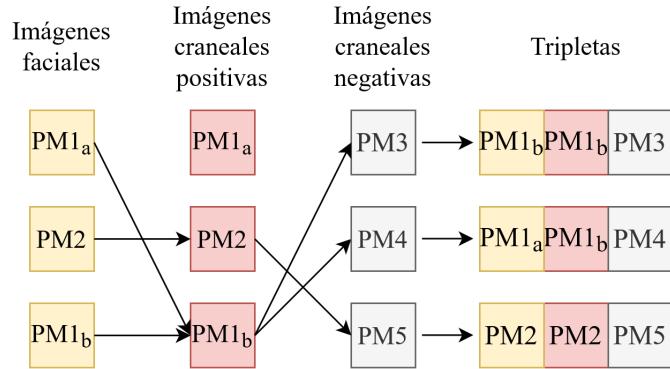


Figura 5.14: La estrategia generación online de tripletas *batch-hard* consiste en emparejar cada imagen facial con la imagen craneal positiva (del mismo individuo) y la negativa (de otro individuo diferente) más complicadas, a mayor y menor distancia de la facial respectivamente. En el ejemplo, tanto la imagen facial (amarillo) PM1_a como la PM1_b son emparejadas con la imagen craneal positiva (rojo) PM1_b dado que es la más complicada en ambos casos; no obstante, PM1_a se empareja con la imagen craneal negativa (gris) PM4 y la PM1_b con la PM3 por ser correspondientes con las más difíciles en sus casos. Para la imagen facial PM2 el procedimiento es idéntico, con la salvedad de que solo se dispone de una imagen craneal del mismo individuo y, por tanto, se empareja directamente con esta.

tidumbre pero que sí reduzca notablemente la muestra de candidatos. Un eventual error grande (por ejemplo, que el modelo prediga que un 80 % de las imágenes faciales son más parecidas al cráneo que la imagen facial real) no permitiría reducir la muestra de candidatos. Dado que la MSE penaliza en mayor medida los errores más grandes, se plantea como una función de pérdida adecuada para el problema.

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (5.20)$$

Mientras que esta arquitectura no tienen en cuenta de forma simultánea los ejemplos positivos y los negativos, como hace la SNN entrenada con TL, la salida de la red es más intuitiva y fácil de aplicar a la hora de hacer predicciones. En este caso, para hacer una predicción basta con alimentar la SNN con la imagen facial y la craneal para que devuelva el valor de semejanza entre 0 y 1. La Figura 5.17 ilustra el proceso de predicción de imágenes craneales.

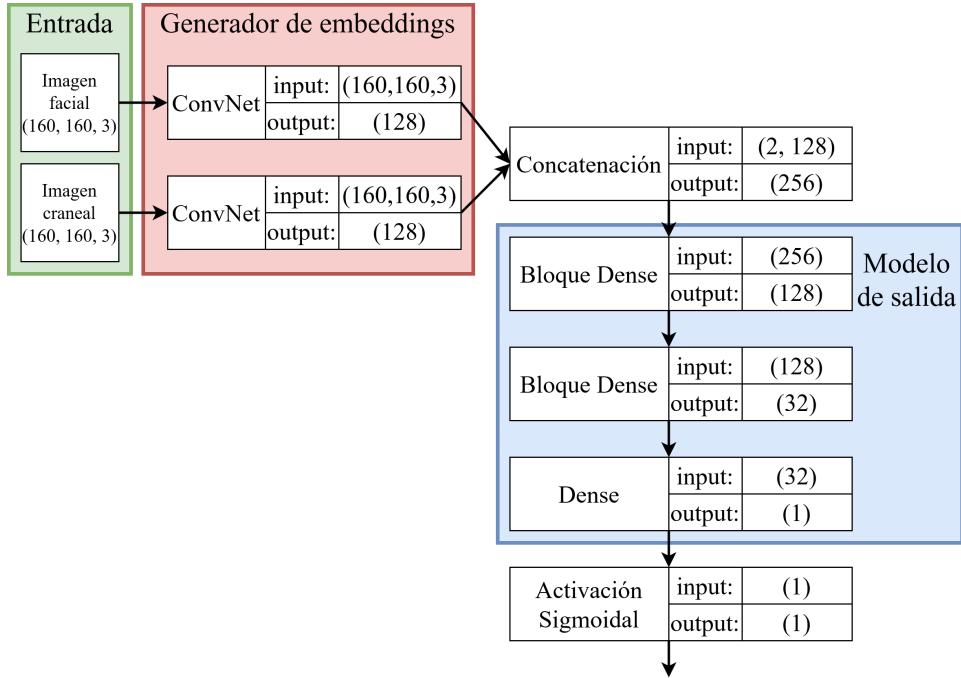


Figura 5.15: Arquitectura de SNN propuesta. Se utiliza *FaceNet* para generar los *embeddings* de las dos imágenes de entrada, una facial y otra craneal. Ambos *embeddings* son concatenados y alimentan a un modelo de salida compuestos por dos *Bloques Dense* (ver Figura 5.16), conectados a una capa Densa que genera una salida sigmoidal. Elaboración propia.

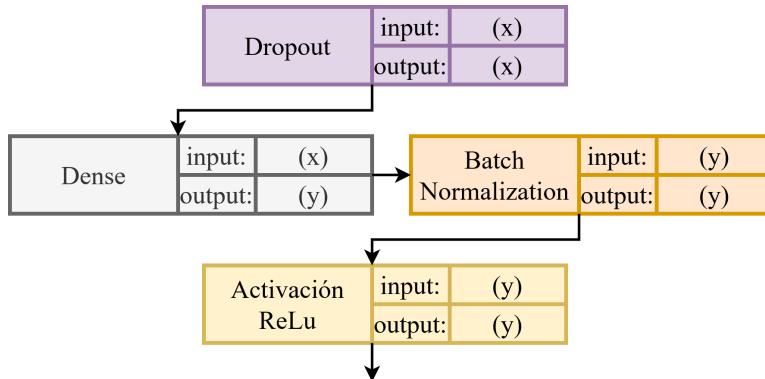


Figura 5.16: Bloque de capas propuesto y utilizado en el modelo de salida. Recibe una entrada de dimensión x , conectada a una capa Densa con *Dropout* con y neuronas cuya salida es, a su vez, conectada a una capa de *Batch Normalization* con una activación ReLU. La salida del bloque tiene dimensionalidad y . Elaboración propia.

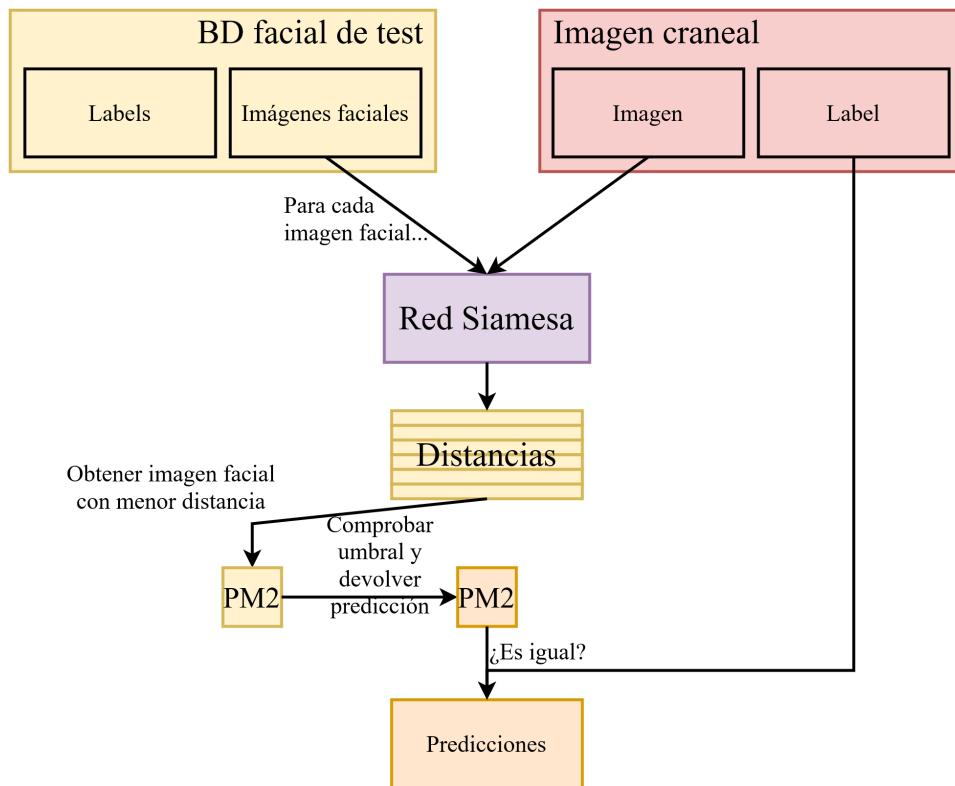


Figura 5.17: Proceso de predicción de las imágenes craneales en las SNN con salida sigmoidal. La imagen craneal es emparejada con cada una de las imágenes faciales de la BD, alimentando la Red Neuronal con todos los pares, sucesivamente, y obteniendo una distancia entre cada imagen craneal y cada imagen facial. La imagen facial que mayor distancia obtiene es la predicha, siempre que esta distancia sea mayor al umbral de decisión (ver Sección 5.2.3.3).

5.2.3.3. Umbral de decisión

En general, el objetivo de los modelos de IDH es comparar la información disponible de la víctima con la información de los individuos de una base de datos y devolver el individuo más parecido. Por ello, y como se ha mencionado varias veces anteriormente, el objetivo de los modelos elaborados en este TFG es recibir una imagen craneal y establecer una correspondencia con una de las imágenes faciales de una base de datos. Al tratarse de un escenario de la vida real, existen dos posibles situaciones al recibir una imagen craneal:

- La imagen facial asociada **no** está disponible en la BD. En este caso, la respuesta del modelo debe ser rechazar la identificación, es decir, no devolver ninguna imagen facial como correspondiente a dicho cráneo.

- La imagen facial asociada **sí** está disponible en el BD. El modelo debe devolver la imagen facial correspondiente a dicho cráneo: la imagen facial de la misma persona.

Para conseguir replicar las anteriores situaciones mediante los modelos de DL se ha definido un **umbral de decisión**, β , que determina cuando una imagen craneal y una imagen facial son lo suficientemente similares como para establecer una correspondencia entre ambas. Si una imagen facial es comparada con una craneal y no cumplen el umbral de decisión, entonces no pueden pertenecer a la misma persona: si ninguna imagen facial de la BD cumple el umbral de decisión, entonces la identificación es rechazada y no se devuelve ninguna imagen facial.

A pesar de que el umbral de decisión es mencionado en el estudio de la identificación facial, por ejemplo en [105], no se establece en ningún caso un algoritmo para su cómputo. Por ello, en este TFG se ha explorado la aplicación del umbral de decisión al problema en cuestión y a los modelos de DL propuestos. En el caso de la Red Neuronal Siamesa con Triplet Loss, si la distancia entre dos imágenes es **menor** al umbral entonces se consideran suficientemente similares, sino no pueden ser emparejadas. En el caso de la Red Neuronal Siamesa con salida sigmoidal, si la salida de la red (la semejanza, entre 0 y 1) para dos imágenes es **mayor** que el umbral entonces son suficientemente semejantes, sino no pueden ser emparejadas.

El valor del umbral de decisión ha de ser determinado al final del entrenamiento de la red neuronal, en función de las distancias entre las imágenes de los pares positivos y las de las imágenes de los pares negativos. En este trabajo se proponen tres métodos de obtención del umbral de decisión, el algoritmo TA-TR, el algoritmo RAP-RAN y el algoritmo de árbol de decisión. Se detallan a continuación.

Algoritmo TA-TR

Este algoritmo está basado en el concepto de *accuracy* y está definido por el número de *True Accepts* (TA, número de pares positivos clasificados como semejantes con el umbral de decisión) y por el número de *True Rejects* (TR, número de pares negativos clasificados como no semejantes con el umbral de decisión). El umbral de decisión TA-TR óptimo es aquel que maximiza la suma de ambos ratios. El procedimiento para obtener es el siguiente:

1. Recibir las distancias D_P y D_N entre pares de imágenes positivos y pares de imágenes negativos, respectivamente.
2. Considerar como posibles umbrales cada uno de los elementos de D_P y D_N .
3. Calcular el número de TA y de TR con cada umbral y obtener la suma de ambos.

4. Devolver el umbral β que maximiza la suma de TA y TR.

Algoritmo RAP-RAN

Este método se basa en la construcción de dos histogramas: uno para las distancias de pares positivos y otro para las de los pares negativos (ver Figura 5.18). Se computa el Ratio de Área debajo de los Positivos (RAP) y el Ratio de Área debajo de los Negativos (RAN): la distribución de barras para el cálculo de estos ratios se puede ver en la Figura 5.19.

- El RAP es el área debajo de las barras de los pares positivos cuya distancia permite considerar las imágenes como semejantes por el umbral (en el caso de SNN con TL la distancia es menor al umbral y en el caso de SNN con salida sigmoidal la distancia es mayor al umbral) en proporción al área total de las barras de pares positivos.
- El RAN es el área debajo de las barras de los pares negativos cuya distancia permite considerar las imágenes como diferentes por el umbral (en el caso de SNN con TL la distancia es mayor al umbral y en el caso de SNN con salida sigmoidal la distancia es menor al umbral) en proporción al área total de las barras de pares negativos.

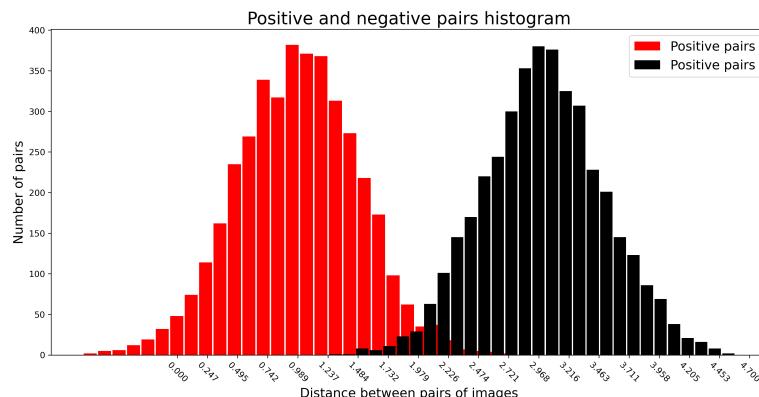


Figura 5.18: Histogramas de la distribución de distancias entre imágenes de pares positivos (en rojo) y entre imágenes de pares negativos (en negro). Este ejemplo ha sido generado con datos no reales (con el objetivo de establecer una visualización clara) y se corresponde con un escenario en el que pares de imágenes semejantes están a menor distancia. Elaboración propia.

El umbral RAP-RAN óptimo es aquel que maximiza el producto de la métrica RAP por la RAN. El método RAP-RAN es el siguiente:

1. Recibir las distancias D_P y D_N entre pares de imágenes positivos y pares de imágenes negativos, respectivamente.

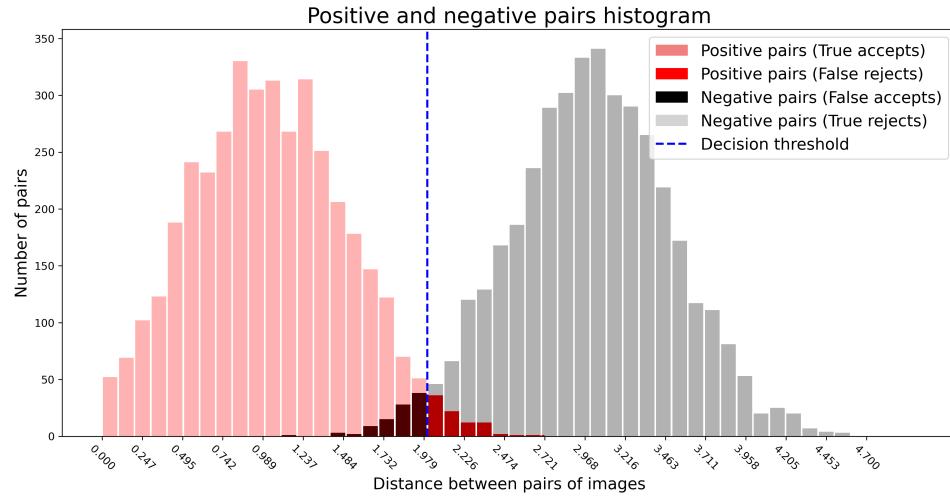


Figura 5.19: El umbral de decisión divide ambos histogramas de distancias (entre pares positivos y entre pares negativos) en dos. Para este ejemplo se han utilizado datos no reales: los pares positivos con distancia menor al umbral (en azul, línea discontinua) están correctamente clasificados (rojo claro) y los que tienen mayor distancia están mal clasificados (rojo oscuro), mientras que los pares negativos con distancia mayor que el umbral están bien clasificados (gris claro) y los que tienen menor distancia están incorrectamente clasificados (negro).

2. Considerar como posibles umbrales cada uno de los elementos de D_P y D_N .
3. Crear los histogramas positivo y negativo.
4. Computar las métricas RAP y RAN para cada umbral y obtener el producto de ambas.
5. Devolver el umbral β que maximiza el producto de RAP por RAN.

Algoritmo de árbol de decisión

Este tercer algoritmo utiliza un árbol de decisión para decidir si dos imágenes son suficientemente similares como para ser emparejadas: un árbol de decisión es una solución gráfica (basada en un grafo) a un problema (ver Figura 5.20). Se pueden utilizar para la clasificación binaria (en este caso, clasificar un par de imágenes en semejante o diferente) y, utilizando un único nodo de decisión, permiten definir si la entrada pertenece a una de las clases si es menor que un umbral y a la otra si es mayor. El umbral de decisión es aprendido durante el entrenamiento del árbol que, en este TFG se realiza con las distancias entre pares positivos y las distancias entre pares negativos.

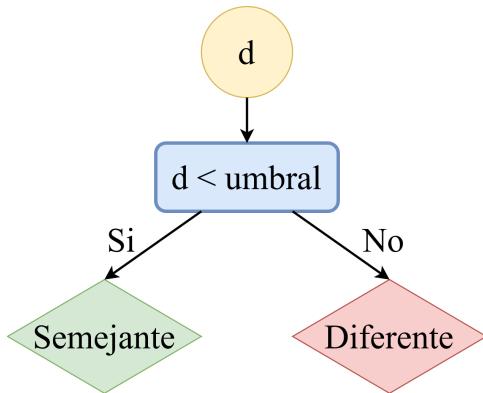


Figura 5.20: Ejemplo de árbol de decisión para la elección del umbral de decisión en el escenario de una SNN con TL. El árbol recibe la distancia entre un par de imágenes y en el nodo de decisión la compara con el umbral del nodo: si es menor entonces son semejantes y si es mayor son diferentes. Elaboración propia.

El procedimiento seguido para obtener el umbral de decisión del árbol es el siguiente:

1. Recibir las distancias D_P y D_N entre pares de imágenes positivos y pares de imágenes negativos, respectivamente.
2. Crear un árbol con un único nodo de decisión.
3. Entrenar el árbol para aprender un umbral de decisión que separe los pares en semejantes o diferentes en función de su distancia.
4. Devolver el umbral β aprendido por el árbol.

5.2.4. Modelos propuestos

Esta subsección recoge las propuestas realizadas en este TFG: cinco modelos para el problema de la IDH directa basada en imágenes faciales y craneales. A continuación se listan estos modelos, junto con una descripción de los hiperparámetros ligados a sus arquitecturas.

Primera propuesta: SNNTLBASIC

Consiste en una SNN con tres sub redes, entrenada con TL y empleando la Estrategia de generación offline de tripletas, propuesta en este trabajo y descrita previamente. Las tripletas son generadas independientemente de su calidad, esto es, sin seleccionar de ninguna forma las tripletas *hard* y *semi-hard*. Este modelo consta de los siguientes hiperparámetros:

1. α , margen de la TL.
2. λ , regulador de la regularización L2 propuesta para la TL.

Segunda propuesta: SNNTLSEL

Es una adaptación del modelo SNNTLBASIC: de nuevo se trata de una SNN, entrenada con TL empleando la Estrategia de generación offline de tripletas. La novedad que se introduce es una selección de tripletas durante la generación de las mismas, descartando aquellas con calidad deficiente. Solo son utilizadas las tripletas *hard* y *semi-hard* para computar la función de pérdida. Los hiperparámetros son idénticos a los del modelo SNNTLBASIC.

1. α , margen de la TL.
2. λ , regulador de la regularización L2 propuesta para la TL.

Tercera propuesta: SNNTLC

Este modelo integra la propuesta de generación offline de tripletas, la Estrategia de generación offline de tripletas, con la TLC detallada en [108]. Una vez más, es una SNN con tres sub redes, cuyos hiperparámetros son:

1. α , margen de la TLC.
2. λ , regulador de la regularización L2 propuesta para la TLC.
3. α_{pen} , regulador de la recompensa o penalización de la TLC.
4. ϵ , margen de las mejores tripletas TLC.

Cuarta propuesta: SNNTLONLINE

Este modelo es una SNN con tres sub redes, entrenada con TL y empleando, o bien la Estrategia de generación online de tripletas batch-all, o bien la Estrategia de generación online batch-hard de tripletas, ambas propuestas en este TFG. Los hiperparámetros de esta arquitectura son:

1. α , margen de la TL.
2. λ , regulador de la regularización L2 propuesta para la TLC.
3. T_B , flag indicando la estrategia de generación de tripletas a utilizar (B_A , *batch-all* o B_H , *batch-hard*).

Quinta propuesta: SNNSIGMOID

Este último modelo consiste en la SNN con salida sigmoidal propuesta en este TFG (ver Sección 5.2.3.2), entrenada con MSE. Sus hiperparámetros son los siguientes:

1. R_D , ratio de *dropout* de las capas de *dropout* de los bloques Dense del modelo de salida.
2. λ , regulador de la regularización L2 de las capas densas de los bloques Dense del modelo de salida.

5.2.4.1. Hiperparámetros comunes a todos los modelos

Dado que los cinco anteriores modelos propuestos están fundamentados en SNNs, comparten ciertos hiperparámetros entre ellos.

- Hiperparámetros del optimizador Adam. Son tres: el *learning rate*; β_1 , el ratio de decaimiento exponencial para el primer momentum; y β_2 , el ratio de decaimiento exponencial para el segundo momentum. No se entra en el detalle de estos dos últimos, ya que han sido fijados a los definidos en [64].
- Bloques de capas a reentrenar de *FaceNet*.
- Número de épocas de entrenamiento.
- Tamaño de los batches de entrenamiento.

Capítulo 6

Experimentación

Este capítulo está orientado a exponer los diferentes detalles de la experimentación realizada en este TFG. En primer lugar, se expone el protocolo de validación utilizado, la separación del conjunto de datos en entrenamiento, validación y test realizada y el formato de test de los modelos, junto con las métricas utilizadas para ello. A continuación se muestra el preprocesado realizado sobre el conjunto de modelos craneales e imágenes faciales. Finalmente, es presentan los diferentes experimentos realizados, así como la discusión de los resultados obtenidos.

6.1. Protocolo de validación experimental

En este trabajo se emplea el protocolo de validación **Validación Cruza-d**a (*Cross Validation*, CV) para medir de forma cuantitativa el rendimiento de los modelos, más en concreto CV con 5 *folds*. Este método consiste en dividir el conjunto de entrenamiento en 5 subconjuntos (*folds*) cuyas características sean lo más parecidas entre sí (en el sentido de número de ejemplos presentes en cada *fold*, mismo ratio de ejemplos de cada clase, etc.) y realizar 5 entrenamientos independientes del modelo. En cada una de las 5 iteraciones se utilizan 4 *folds* como el conjunto de entrenamiento y el *fold* restante como conjunto de test, obteniendo las métricas de error en este último (ver Figura 6.1). Adicionalmente, se ha utilizado un **conjunto de datos de validación**: un conjunto similar al de test utilizado para medir el rendimiento del modelo durante el entrenamiento. Durante CV, el conjunto de validación es separado de los *folds* de entrenamiento.

Los conjuntos de entrenamiento y test son delimitados de forma estocástica, por lo que la validación **hold-out** (utilizando una única partición en entrenamiento, validación y test, ver Figura 6.2) podría llevar a disponer de unos resultados que, debido al componente aleatorio, no sean adecuados. Por ello, la CV permite evaluar de una forma más robusta los modelos, especialmente en los casos en los que se dispone de una escasa cantidad de datos

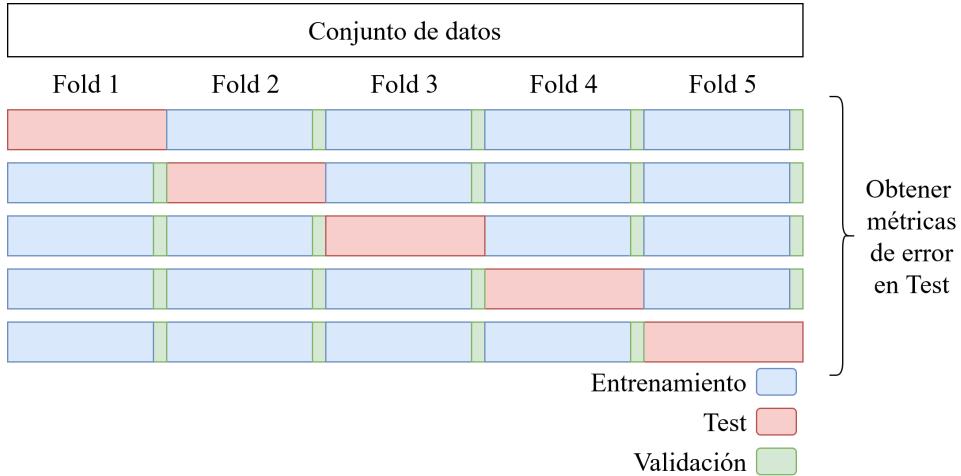


Figura 6.1: Protocolo de CV con 5 *folds*: en cada iteración uno de los *folds* es utilizado como test y los 4 restantes como entrenamiento y validación. Cada modelo se entrena un total de 5 veces independientes. Elaboración propia.

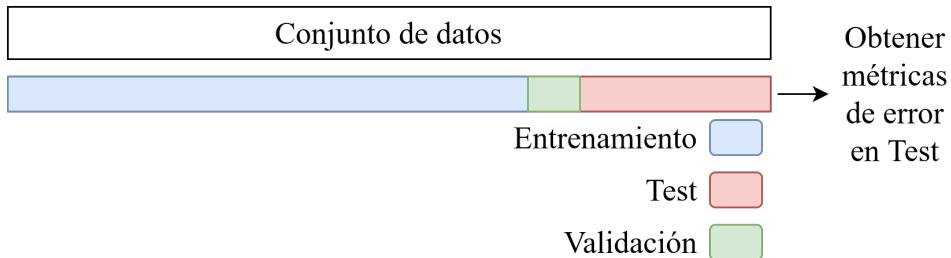


Figura 6.2: Protocolo de hold-out: el conjunto de datos es únicamente dividido en entrenamiento, validación y test una única vez. Cada modelo es entrenado solo una vez. Elaboración propia.

y una división de estos en entrenamiento y test podría no ser representativa. En cada iteración de entrenamiento, los datos de entrenamiento y test son diferentes, por lo que las métricas obtenidas muy probablemente variarán.

CV 5-*fold* tiene la contrapartida de ser un método costoso computacionalmente comparado con la validación por hold-out debido a que cada modelo es entrenado y evaluado cinco veces. Por ello, hold-out puede ser una técnica interesante en escenarios donde el coste computacional de los entrenamientos sea elevado, como es el caso de las SNNs. Como se detallará en la Sección 6.5.2, debido a la escasez de estudios previos en la materia, la selección a priori de hiperparámetros en este problema era complicada. Por esta razón, se decidió realizar una batería experimental con hold-out permitiendo una búsqueda y selección de los hiperparámetros más prometedores. Posteriormente, los mejores modelos de hold-out fueron evaluados mediante

CV para obtener unos resultados más fiables.

6.2. Separación en entrenamiento, validación y test

El conjunto de datos consta de 114 individuos, de los cuales 57 son positivos y 57 son negativos (mitad y mitad). Dado que las imágenes 2D craneales de un mismo individuo son generadas a partir del mismo modelo 3D, la separación de los conjuntos no puede realizarse a partir del conjunto de datos de imágenes 2D, sino que debe realizarse sobre los individuos. Además, para mantener los conjuntos balanceados (mismo número de individuos positivos que negativos) se han separado los individuos positivos y los individuos negativos de forma independiente.

En todos los experimentos realizados se ha separado 10 % de los individuos de entrenamiento para el conjunto de validación. Dado que se han realizado dos baterías experimentales, una utilizando hold-out y otra utilizando CV 5-fold, el conjunto de entrenamiento ha sido utilizado de forma diferente durante los experimentos:

- CV 5-fold. El conjunto se separó en 5 *folds* de individuos, tres de ellos formados por 11 individuos positivos y 11 individuos negativos y dos de ellos formados por 12 positivos y 12 negativos. Durante una iteración, un *fold* es utilizado de test y los otros cuatro son utilizados para entrenamiento y validación.
- Hold-out. El conjunto se separó en tres: test (25 % de los individuos, 14 positivos y 14 negativos), validación (10 % del 75 % restante, 4 positivos y 4 negativos) y entrenamiento del hold-out (90 % del 75 % restante, 39 positivos y 39 negativos).

6.3. Test de los modelos y métricas de error

La etapa posterior al entrenamiento en la que se mide el rendimiento de un modelo de DL se denomina test. Consiste en utilizar un conjunto de datos de test (independiente al de entrenamiento¹) para medir en él una métrica de error. El test de los modelos en este trabajo consiste en comparar una serie de imágenes craneales con una **BD de imágenes faciales de test**: cada imagen craneal es identificada con una imagen facial de la BD (la más semejante) siempre que cumpla el umbral de decisión o, sino, la identificación

¹El conjunto de Test debe ser completamente diferente al conjunto de entrenamiento: cualquier ejemplo utilizado en entrenamiento no debe ser utilizado en test. Esto es debido a que la red neuronal ha utilizado los ejemplos de entrenamiento para aprender conocimiento sobre el problema y, si se empleasen en test, la medida del rendimiento de la red estaría sesgada.

es rechazada. La BD de imágenes faciales es un conjunto de imágenes faciales de diferentes individuos no utilizados en entrenamiento.

Dada la naturaleza del problema y con el objetivo de replicar un escenario real, en este trabajo se ha dividido el test en dos etapas: Test Positivo y Test Negativo. El **Test Positivo** es realizado a través de un conjunto de datos de test compuesto de imágenes craneales positivas, esto es, imágenes craneales de individuos para los que se dispone de imágenes faciales. El **Test negativo** consiste en comparar imágenes craneales negativas (imágenes pertenecientes a individuos de los que no dispone de una imagen facial) con las imágenes faciales de la BD de test. Estos dos procesos de evaluación de los modelos replican, respectivamente, la situación en la que se dispone en la BD de la imagen facial perteneciente a la misma persona que el cráneo y el caso en el que no se dispone de dicha imagen. Para ello, se incluyen las imágenes faciales de los individuos de test positivo en la BD de test.

Para evaluar el rendimiento de los modelos a la hora de hacer predicciones positivas de los individuos de test se va a utilizar el *accuracy*: el número de identificaciones correctas en proporción al número de identificaciones totales realizadas. Esta métrica permite cuantificar la calidad de las predicciones en total (Acc , Ecuación 6.1), en Test Positivo (Acc_P , Ecuación 6.2) y en Test Negativo (Acc_N , Ecuación 6.3). En las anteriores ecuaciones TP , FN , TN y FP se corresponden con verdaderos positivos (imágenes craneales positivas correctamente predichas), falsos negativos (imágenes craneales positivas incorrectamente predichas), verdaderos negativos (imágenes craneales negativas correctamente predichas) y falsos negativos (imágenes craneales negativas incorrectamente predichas).

$$Acc = \frac{TP + TN}{TP + FN + TN + FP} \quad (6.1)$$

$$Acc_P = \frac{TP}{TP + FN} \quad (6.2)$$

$$Acc_N = \frac{TN}{TN + FP} \quad (6.3)$$

El *accuracy* es utilizado en relación a la identificaciones positivas. No obstante, como se ha reiterado en numerosas ocasiones a lo largo de este trabajo, en IDH es interesante también reducir la muestra de candidatos. Por esta razón, se utilizan métricas adicionales para medir este aspecto en test positivo y en test negativo.

En cuanto a test positivo, al comparar una imagen craneal con la BD de test se obtiene un nivel de semejanza con cada imagen facial de dicha BD, pudiendo ordenar así las imágenes faciales por su semejanza. El **ranking** de una imagen craneal de test positivo es la posición en la que fue predicha la imagen facial asociada a ella (la imagen facial del mismo individuo). Por

ejemplo, que el ranking de una imagen sea 10 significa que la imagen facial asociada es la décima más semejante según el modelo. El procedimiento completo de obtención de los rankings de las imágenes craneales se puede consultar en la Sección Algoritmo de cómputo de los rankings en Test Positivo del Apéndice 7. Los rankings permiten visualizar en las curvas CMC el grado de reducción de la muestra de candidatos conseguido por el modelo [41]. Para ello, se define el *top-k accuracy* (Acc_k) de una imagen craneal de test positivo como

$$Acc_k = \begin{cases} 1 & \text{si el ranking es menor o igual a } k \\ 0 & \text{en otro caso} \end{cases} \quad (6.4)$$

A nivel de la muestra al completo, el top-k accuracy representa el ratio de imágenes craneales cuya imagen facial aparece entre las primeras k posiciones. La curva CMC muestra la evolución del promedio de Acc_k en la muestra, desde el *top-1 accuracy* hasta el *top-n accuracy*, siendo n el tamaño de la muestra de candidatos: el *top-n accuracy* siempre será 1.0. Se define a partir del concepto de curva CMC la otra métrica de error a utilizar en test positivo: el **tamaño de la muestra reducida** (Acc_k^1), el menor ranking h tal que $Acc_h = 1.0$.

En la Figura 6.3 se puede ver un ejemplo de curva CMC: en este caso, de una muestra de 60 candidatos, el modelo únicamente realiza el correctamente 20 % de las predicciones (el *top-1 accuracy* es 0.2) pero consigue reducir la muestra de candidatos a la mitad, ya que $Acc_k^1 = 30$ (el *top-30 accuracy* es 1.0).

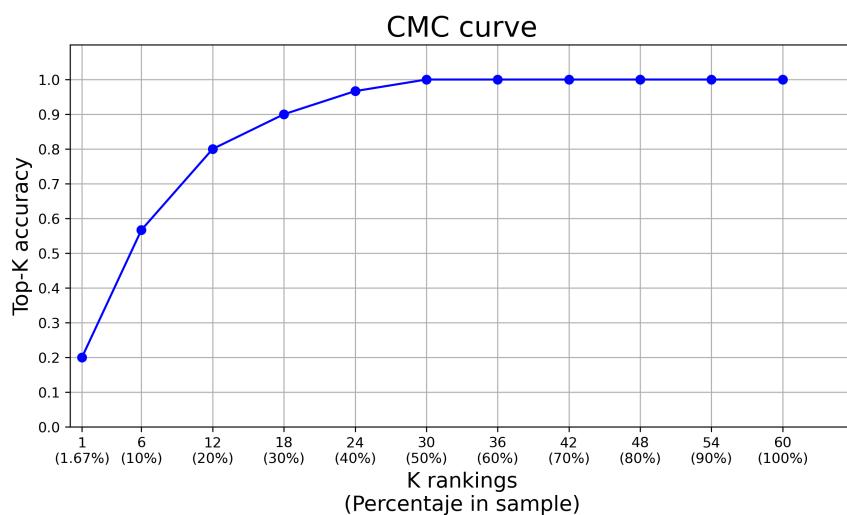


Figura 6.3: La curva CMC permite visualizar en qué medida se ha conseguido reducir la muestra de posibles candidatos, aparte de visualizar también el número de predicciones correctas. Elaboración propia.

Por otro lado, en cuanto a test negativo, una predicción correcta consiste en rechazar todas las imágenes faciales porque ninguna de ellas es lo suficientemente semejante como para ser emparejada². Una predicción incorrecta resulta cuando al menos una imagen facial de la BD es lo suficientemente semejante de acuerdo con el umbral de decisión. Cuantificar el rendimiento de los modelos en este sentido no es una tarea estudiada previamente en este problema ni en reconocimiento facial, por lo que en este TFG se propone una métrica de error para las imágenes craneales negativas: **NFA** (Number of False Accepts). El NFA de una imagen craneal negativa S es el número de imágenes faciales F incorrectamente asociadas a ella: NFA es el número de imágenes faciales cuya distancia a la imagen craneal se considera suficientemente semejante (menor al umbral de decisión en SNN con TL y mayor al umbral en SNN con salida sigmoidal). Este concepto, basado en el hecho de que las imágenes craneales negativas no deben ser asociadas con ninguna imagen facial de la BD de test, se plasma para la SNN con TL y la SNN con salida sigmoidal en las Ecuaciones 6.5 y 6.6, respectivamente.

$$\text{NFA}(S) = \text{card}(F_N), \quad F_N = \{f \in F : d(S, f) < \beta\} \quad (6.5)$$

$$\text{NFA}(S) = \text{card}(F_N), \quad F_N = \{f \in F : d(S, f) > \beta\} \quad (6.6)$$

Observarse que la predicción de una imagen craneal negativa S es correcta si $\text{NFA}(S) = 0$. El procedimiento completo de obtención de los NFA de las imágenes craneales se puede consultar en la Sección Algoritmo de cómputo de los NFA en Test Negativo del Apéndice 7.

Este trabajo propone también el concepto de curva NFA que, de forma similar al de la curva CMC, permite visualizar el número de imágenes faciales incorrectamente clasificadas en Test Negativo. Se define el top- k NFA (NFA_k) de una imagen craneal negativa S como:

$$\text{NFA}_k(S) = \begin{cases} 1 & \text{si } \text{NFA}(S) < k \\ 0 & \text{en otro caso} \end{cases} \quad (6.7)$$

La curva NFA muestra la evolución del promedio de NFA_k en la muestra de test, desde el top-1 NFA hasta el top- n NFA, siendo n el tamaño de la muestra de candidatos: el top- n NFA siempre será 1.0. Nótese que el top-1 NFA se corresponde con la *accuracy* en Test Negativo, ya que es el ratio de imágenes craneales negativas correctamente clasificadas respecto al total de imágenes negativas de test. En la Figura 6.4 se puede ver un ejemplo de curva NFA: en una muestra de 60 candidatos, el modelo rechaza correctamente el 70 % de las imágenes faciales (el top-1 NFA es 0.7) y, como mucho, se aceptan incorrectamente menos de 12 imágenes (el top-12 NFA es 1.0).

²En la SNN con TL dos imágenes no eran suficientemente semejantes si la distancia entre ambas era mayor al umbral de decisión, mientras que en la SNN con salida sigmoidal, al contrario, si la distancia era menor al umbral

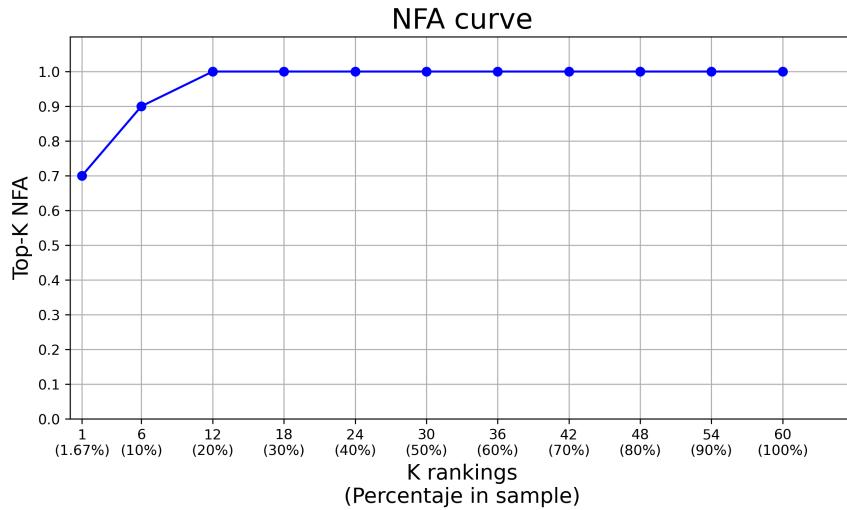


Figura 6.4: La curva NFA permite visualizar el grado de confianza con el que se rechaza una imagen negativa. Elaboración propia.

6.4. Generación de conjuntos de imágenes 2D

En esta sección se presenta el preprocesamiento de los datos realizado: la generación de imágenes 2D craneales a partir de los modelos 3D, la generación de nuevas imágenes faciales mediante *data augmentation* y el diseño de las bases de datos de imágenes faciales utilizadas en validación y test.

6.4.1. Generación del dataset de imágenes 2D craneales

El primer paso a realizar en la experimentación en este TFG ha sido procesar el conjunto de datos de modelos 3D craneales. Cada individuo del *dataset* tiene un identificador único (de PM1 a PM117³): tanto las imágenes craneales generadas como las imágenes faciales han sido etiquetadas con el identificador del individuo correspondiente.

De cara a alinear todos los modelos 3D del *dataset*, se ha seleccionado el individuo PM11 (ver Figura 6.5) como referencia, debido a que los detalles de su modelo 3D permiten realizar el proceso de alineamiento de forma sencilla. En el *dataset* hay 21 mallas ya alineadas correctamente con este cráneo, mientras que las 93 restantes no están alineadas adecuadamente. En las Figuras 6.6 y 6.7 se pueden ver dos ejemplos diferentes del proceso de alineamiento de modelos 3D: dentro de los modelos no alineados del *dataset*, algunos están notoriamente más desalineados que otros.

Tras el alineado, se procedió a eliminar los elementos que causaban im-

³Recordar que los individuos PM24, PM40 y PM106 no son utilizados debido a que sus modelos 3D craneales son incorrectos.



Figura 6.5: Perspectiva frontal del modelo 3D del cráneo del individuo PM11, modelo utilizado para el alineamiento de todos las mallas del conjunto de datos.

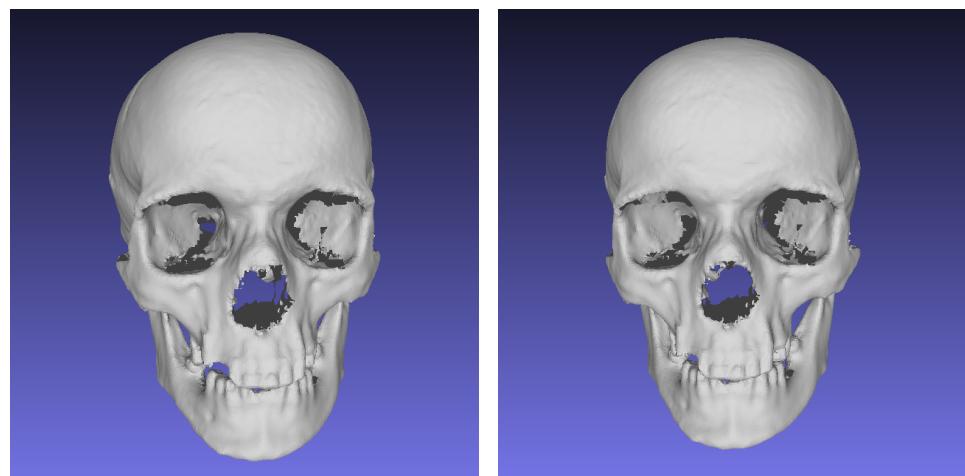


Figura 6.6: Modelo 3D correspondiente al individuo PM1, ligeramente no alineado. En la izquierda se puede ver la malla original y en la derecha la malla alineada con el modelo PM11.

perfecciones en los modelos (soportes, fragmentos de columnas vertebrales, etc.). Se han procesado los 35 modelos que presentan alguno de estos elementos a través de MeshLab para eliminar dichos fragmentos de la malla. En la Figura 6.8 se puede ver un ejemplo del resultado obtenido.

Tras estos dos pasos, se procedió a generar las imágenes 2D a partir de

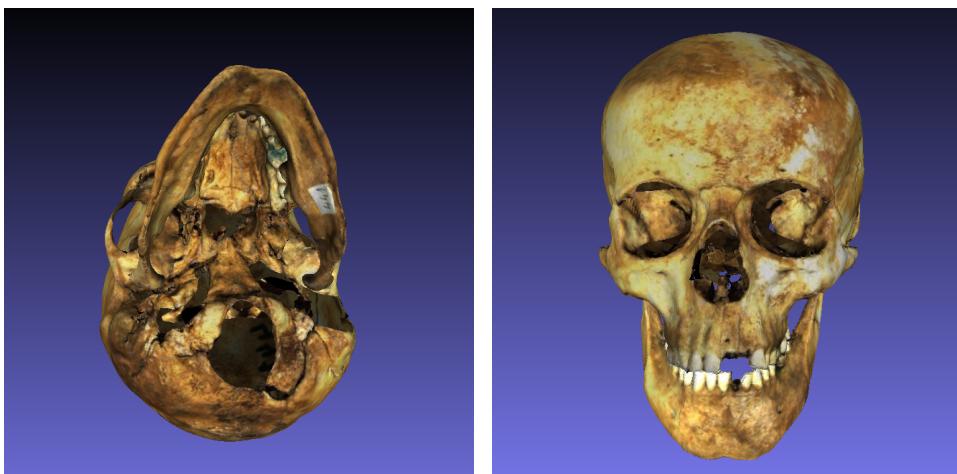


Figura 6.7: Modelo 3D correspondiente al individuo PM32, notoriamente no alineado. En la izquierda se puede ver la malla original y en la derecha la malla alineada con el modelo PM11. A diferencia del modelo PM1 de la Figura 6.6, la malla PM32 está notablemente desalineada respecto al modelo PM11.

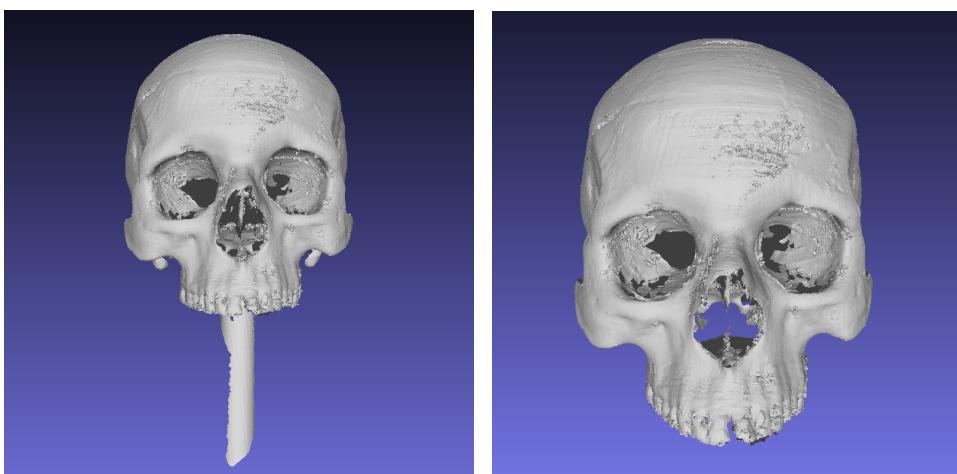


Figura 6.8: Modelo 3D correspondiente al individuo PM73: en la izquierda se puede ver la malla original y en la derecha la malla obtenida tras recortar el soporte que sujeta el cráneo en el modelo original.

los modelos 3D craneales ya procesados. Para ello se implementó un script de Python que automatiza el Algoritmo de generación aleatoria de imágenes, produciendo 100 imágenes 2D diferentes para cada cráneo: en la mitad de ellas se posicionó la cámara a 0.6 metros del cráneo y en la otra mitad a 3 metros. En la Figura 6.9 se muestran varias imágenes 2D generadas para el mismo modelo 3D craneal de entrada.

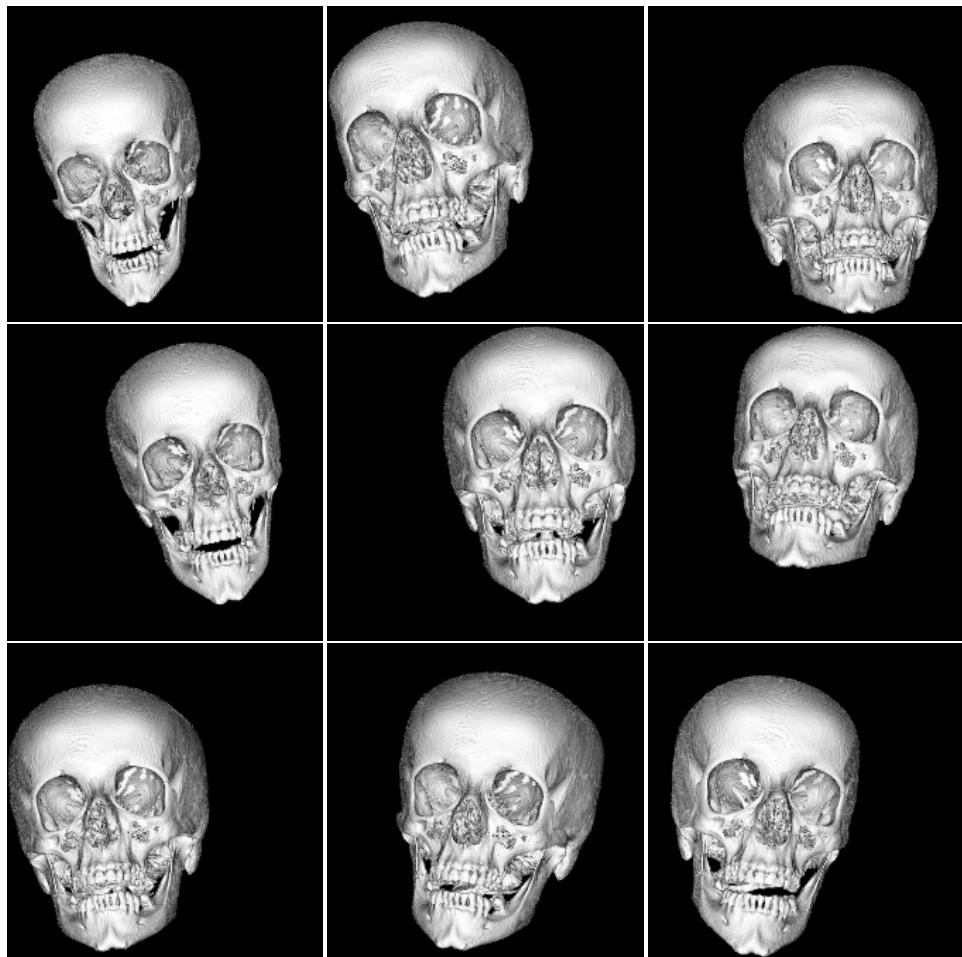


Figura 6.9: Las imágenes 2D son generadas a partir de los modelos 3D craneales tras aplicar rotaciones y desplazamientos aleatorios sobre el modelo y distanciando la cámara de él. En la figura se ven 9 diferentes imágenes generadas para el cráneo del individuo PM100.

Las imágenes 2D craneales generadas son utilizadas tanto en el entrenamiento de los modelos como en validación y en test. Durante el entrenamiento son emparejadas con imágenes faciales para formar tripletas de imágenes (ver Figura 6.10) en la SNN con TL o pares de imágenes (ver Figura 6.11) en la SNN con salida sigmoidal.

6.4.2. Data augmentation de imágenes faciales

Uno de los problemas recursivos en el desarrollo de este TFG es la escasez de datos de entrenamiento. En el caso de las imágenes faciales, para los individuos positivos solo se dispone de una, que es repetida para cada imagen

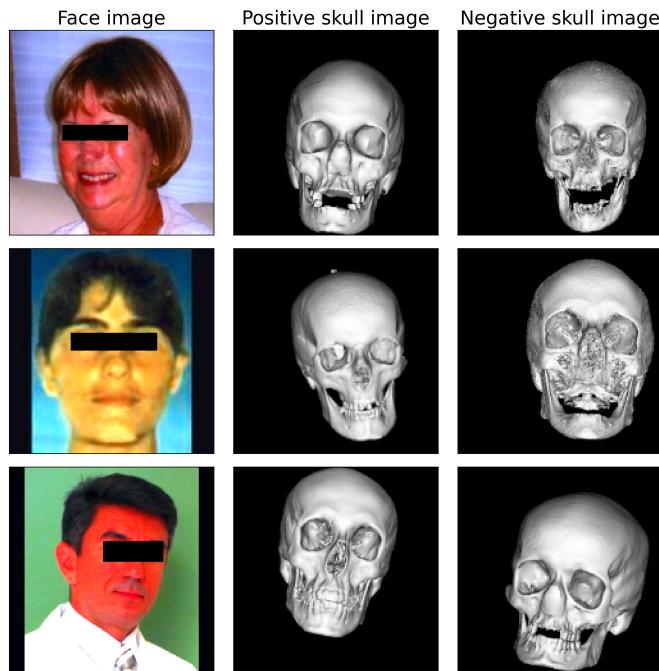


Figura 6.10: Varios ejemplos de tripletas de imágenes. Una imagen craneal positiva es emparejada con su respectiva imagen facial (la perteneciente al mismo individuo) y con una imagen craneal de otro individuo diferente.

2D craneal de ese individuo. Para aumentar la varianza de los datos se ha implementado un *data augmentation* offline (antes del entrenamiento del modelo), consistente en las siguientes transformaciones aleatorias:

- Volteos horizontales aleatorios.
- Ajustes aleatorios del brillo de la imagen.
- Ajustes aleatorios del contraste de la imagen.
- Ajustes aleatorios de la saturación de la imagen.

Este *data augmentation* se ha aplicado únicamente sobre las imágenes faciales de los individuos positivos, ya que para los individuos negativos (en el caso de la SNN con salida sigmoidal) se dispone de las imágenes faciales del *dataset* UTKFace. En la Figura 6.12 se pueden ver varios ejemplos del resultado obtenido.



Figura 6.11: Ejemplos de pares de imágenes faciales-craneales. Las imágenes craneales positivas son emparejadas con su respectiva imagen facial (la perteneciente al mismo individuo), mientras que las imágenes craneales negativas son emparejadas con una imagen facial de otro individuo diferente (seleccionada aleatoriamente del *dataset* UTKFace).

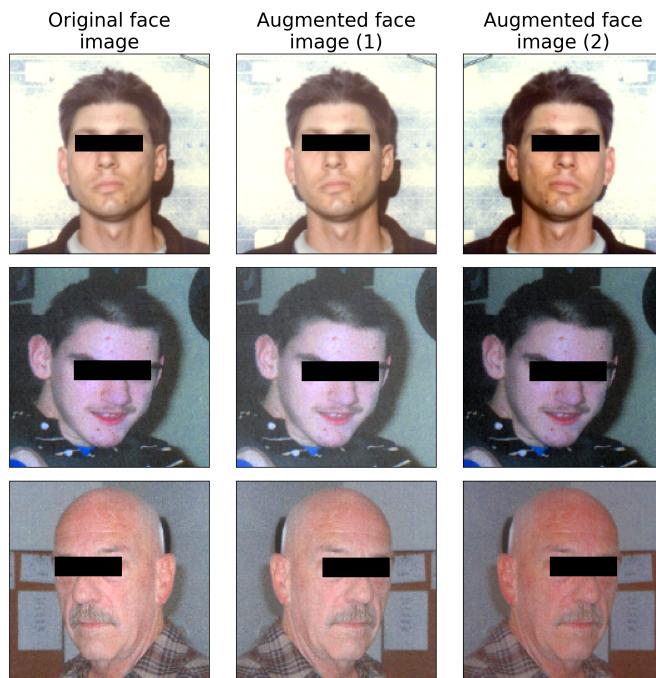


Figura 6.12: Ejemplos del *data augmentation* realizado sobre las imágenes faciales de los individuos positivos.

6.4.3. Generación de la Base de Datos de imágenes faciales de Test

La BD de imágenes faciales de Test debe incluir las imágenes faciales de los individuos positivos de test. El resto de imágenes contenidas en la BD han de ser diferentes entre sí y diferentes a las imágenes faciales utilizadas durante el entrenamiento de la red. Se ha decidido utilizar una BD con 100 imágenes faciales, entre las que se incluyen las de los individuos positivos de test.

Dada la naturaleza de los experimentos de CV 5-fold, es necesario disponer de una BD diferente para cada fold: cada BD debe incluir las imágenes faciales positivas de ese fold, utilizado para test, pero no las imágenes faciales utilizadas para entrenamiento de los restantes folds. Por ello, se ha diseñado una BD para cada uno de los folds, compuesta por las imágenes faciales de los individuos positivos de ese fold, las de los individuos de validación del resto de folds y completada (hasta llegar a 100 imágenes) con imágenes faciales seleccionadas aleatoriamente del *dataset* UTKFace. La Figura 6.13 refleja la construcción de la BD facial de test en CV.

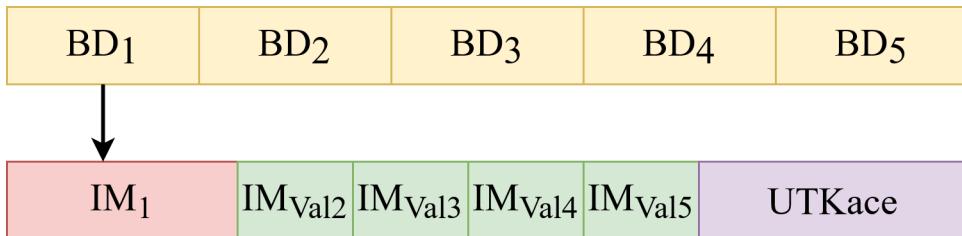


Figura 6.13: Dada la naturaleza de los experimentos de CV, es necesario disponer de una BD facial diferente para cada fold. Por ejemplo, para el fold 1 (IM₁), la BD (BD₁) contiene todas las imágenes faciales positivas del fold 1, las imágenes faciales de validación de los folds 2, 3, 4 y 5 (IM_{Val2}, IM_{Val3}, IM_{Val4} y IM_{Val5}) y es completada hasta las 100 imágenes con el parte del *dataset* UTKFace.

En el caso de hold-out, se diseñó una BD con las imágenes faciales de los individuos positivos separados para Test, los individuos de validación y completada con imágenes seleccionadas aleatoriamente de UTKFace (ver Figura).

Tanto en CV como en hold-out se incluyen las imágenes faciales de validación en las respectivas BD faciales para utilizar las BD durante el entrenamiento para medir el rendimiento en validación. Como estas imágenes no son utilizadas para entrenar el modelo, se pueden utilizar también en el Test del modelo.

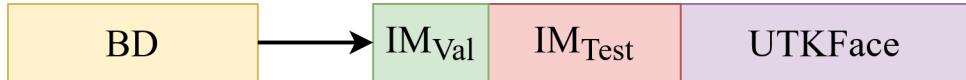


Figura 6.14: La BD facial de test en hold-out está compuesta por las imágenes faciales positivas de test (IM_{Test}), las imágenes faciales de validación (IM_{Val}) y completada hasta las 100 imágenes con parte del *dataset* UTKFace.

6.5. Experimentación con los modelos propuestos

Durante el desarrollo de la experimentación de los modelos de DL ha sido necesario tomar múltiples decisiones relativas al diseño de los modelos y su entrenamiento. Estas decisiones se han tomado en función de los conocimiento adquiridos en las asignaturas *Aprendizaje Automático* y *Visión por Computador*, de las técnicas estado del arte y de los propios resultados experimentales. La Figura 6.15 refleja el proceso final de entrenamiento y test de los modelos.

6.5.1. Elección del modelo base y optimizador

Tanto la arquitectura de SNN con TL como la de SNN con salida sigmoidal están compuestas por una sub red convolucional replicada tres y dos veces respectivamente. En el comienzo de la investigación se utilizó *ResNet50* [45] con la finalidad de desarrollar de forma primitiva el flujo de trabajo, ya que este modelo está disponible preentrenado en la librería TensorFlow. No obstante, tras realizar un análisis de estado del arte en reconocimiento facial con DL (problema similar al afrontado) se decidió utilizar la arquitectura FaceNet debido a que el conocimiento preentrenado de esta red es más específico para este problema.

El modelo de *FaceNet* utilizado [117] fue preentrenado en el conjunto de datos CASIA-WebFace, compuesto por 453453 imágenes de más de 10575 identidades diferentes, utilizando el optimizador Adam [64] para ello. Este modelo en concreto fue testeado en el conjunto LFW, con una *accuracy* de 0.99650 ± 0.00252 . Su arquitectura está detallada en la Sección 5.2.2.

Dado que el modelo base utilizado ha sido entrenado utilizando a Adam y que este optimizador es el propuesto en el estado del arte de reconocimiento facial utilizando SNNs [105], se ha decidido utilizar Adam como algoritmo optimizador del entrenamiento en los experimentos de este TFG.

6.5.1.1. Early Stopping

El *Early Stopping* es una técnica de regularización aplicada a los modelos durante el entrenamiento, cuyo objetivo es reducir el overfitting en el conjunto de entrenamiento regularización [38, 68, 103]. Consiste en monitorizar

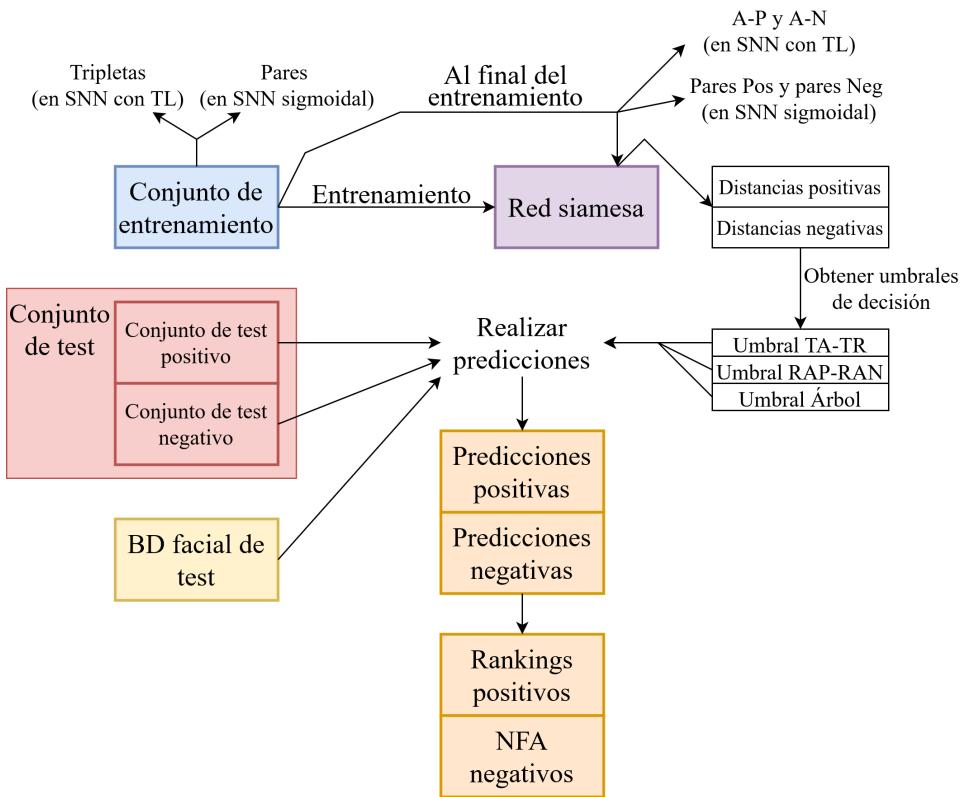


Figura 6.15: Flujo de trabajo seguido en el entrenamiento y test de las SNNs. En la SNN con TL las tripletas están formadas de imágenes faciales (A), imágenes craneales positivas (P) e imágenes craneales negativas (N), mientras que en la SNN con salida sigmoidal los pares son imágenes faciales emparejadas con imágenes craneales positivas (Pares Pos) e imágenes craneales negativas (Pares Neg).

una métrica de error sobre el conjunto de validación al final de cada época y detener el entrenamiento cuando la métrica haya dejado de mejorar.

Debido a la escasa cantidad de datos disponibles para este problema, se ha decidido implementar *Early Stopping* para evitar un posible sobreajuste a los datos de entrenamiento. La métrica utilizada para monitorizar el desempeño del modelo en validación es el tamaño de la muestra reducida, Acc_k^1 , definida previamente. No obstante, de cara a tener independencia del tamaño de la BD de test, se monitoriza el porcentaje de reducción de la muestra, es decir, $\%Acc_k^1 = \frac{Acc_k^1}{tam_f}$, siendo tam_f el tamaño total de la BD de test. Por ejemplo, si $tam_f = 100$ y $Acc_k^1 = 40$ (ningún ranking de validación es mayor que 40), entonces $\%Acc_k^1 = 0.4$: la muestra se ha reducido un 40 %.

El *Early Stopping* implementado en este TFG computa $\%Acc_k^1$ sobre validación al final de cada época y detiene el entrenamiento si no ha mejorado

durante tres épocas seguidas: si $\%Acc_k^1$ no ha disminuido al menos 0.005. Esto equivale a que el modelo no haya conseguido reducir la muestra en un 0.5% durante tres épocas. Por último, destacar que se restauran los mejores pesos del modelo, es decir, los pesos que tenía el modelo al final de la última época en la que mejoró el $\%Acc_k^1$.

6.5.2. Selección de hiperparámetros en hold-out

Ya se ha remarcado la ausencia de estudios en el ámbito del DL en el problema de la IDH con imágenes 2D faciales y craneales. Si bien es cierto que *FaceNet* fue entrenada originalmente utilizando $\alpha = 0.2$ [105], en este TFG se proponen varios modelos que no han sido estudiados anteriormente y el problema en cuestión es diferente, por lo que el comportamiento de los hiperparámetros podría no ser el mismo al del entrenamiento original de *FaceNet*. Por esta razón, se decidió realizar una batería experimental de búsqueda de hiperparámetros para reducir el número de modelos a considerar en CV.

Debido a que el entrenamiento de los modelos presentados en este TFG es costoso, realizar una búsqueda de hiperparámetros mediante CV 5-fold supondría un volumen de cómputo excesivamente grande dada la gran cantidad de combinaciones de hiperparámetros a probar. Esta razón llevó a diseñar esta batería experimental con hold-out en vez de CV 5-fold: con ello se pretende seleccionar los modelos potencialmente mejores para realizar posteriormente unas pruebas más fiables con CV. La Tabla 6.1 recoge los valores de hiperparámetros utilizados específicos de cada modelo. Las decisiones tomadas en relación a los hiperparámetros comunes a todos los modelos son las siguientes:

- Hiperparámetros de Adam. Se ha experimentado con los *learning rates* $10^{-3}, 10^{-4}, 10^{-5}$ y 10^{-6} . Tanto β_1 como β_2 han sido fijados a sus valores por defecto (0.9 y 0.999, respectivamente).
- Bloques reentrenables de *FaceNet* (*fine-tuning* de *FaceNet*). Se ha experimentado reentrenando las siguientes combinaciones de capas (listadas de más profunda a menos profunda):
 - *fully conn* (de ahora en adelante, FC)
 - FC, *inception 5b* (de ahora en adelante, I5b)
 - FC, I5b, *inception 5a* (de ahora en adelante, I5a)
 - FC, I5b, I5a, *inception 4e* (de ahora en adelante, I4e)
- Épocas de entrenamiento. Gracias a la implementación de *Early Stopping*, el número de épocas no es relevante, ya que se entrena el modelo hasta que su rendimiento deje de aumentar.

- Tamaño de *batch*, tam_{batch} . Se ha fijado a 64 para SNNTLBASIC, SNNTLSEL, SNNTLC y SNNSIGMOID. En el caso de SNNONLINE, por las estrategias de generación online de tripletas que utiliza, se recomienda utilizar tamaños de *batch* del orden de mil [105]. No obstante, por motivos técnicos del espacio de memoria, se ha tenido que fijar al mayor tamaño posible, 256.

| Arquitectura | HP | Valores |
|--------------|----------------|-----------------------|
| SNNTLBASIC | α | [0.2, 1, 5, 10, 20] |
| | λ | [1.e-4, 1e-5, 1e-6] |
| SNNTLSEL | α | [0.2, 1, 5, 10, 20] |
| | λ | [1.e-4, 1e-5, 1e-6] |
| SNNTLC | α | [0.2, 1, 5, 10, 20] |
| | λ | [0.01, 0.1, 0.2, 0.4] |
| | α_{pen} | [0.1, 0.2, 0.5, 1] |
| | ϵ | [0.1, 0.5, 0.9] |
| SNNONLINE | α | [0.2, 1, 5, 10, 20] |
| | λ | [0.01, 0.1, 0.2, 0.4] |
| | T_B | $\{B_A, B_H\}$ |
| SNNSIGMOID | R_D | [0.5, 0.7] |
| | λ | [0, 0.1] |

Tabla 6.1: Parrilla de valores de cada hiperparámetro para cada uno de las arquitecturas analizadas. Se muestra para cada tipo de arquitectura propuesta sus hiperparámetros (HP) y los valores experimentados.

Se ha realizado una prueba en hold-out de cada uno de los modelos propuestos en todas las combinaciones de hiperparámetros posibles, combinando tanto los propios de la arquitectura como los comunes a todas, a excepción del tamaño de *batch* (característico de cada arquitectura).

De cara a seleccionar los mejores modelos de cada tipo de arquitectura para evaluarlos con CV 5-fold, se ha tenido en cuenta tanto el *accuracy* total del modelo (*Acc*) como el tamaño de la muestra reducida (Acc_k^1). Se han seleccionado los tres mejores modelos respecto a ambas métricas de cada una de las arquitecturas (seis modelos de cada arquitectura en total). Estos modelos se consideran los potencialmente mejores y serán evaluados en CV 5-fold. La Tabla 6.2 muestra los modelos seleccionados y sus respectivas métricas.

Tabla 6.2: Resultados del hold-out: se recogen los tres mejores modelos en cuanto a *accuracy* en test positivo y negativo (mayor *Acc*) y los tres mejores en cuanto al tamaño de la muestra reducida (menor Acc_k^1) para cada arquitectura. Se destacan en verde los mejores modelos en cuanto a *Acc* y en azul los mejores en cuanto a Acc_k^1 . En la tabla se muestra la combinación de hiperparámetros del modelo, HP, el *learning rate* del optimizador del entrenamiento y las capas de *FaceNet* reentrenadas para cada modelo. El significado de los hiperparámetros de los modelos puede comprobarse en la Sección 5.2.4 y el de las capas de *FaceNet* en la Sección 6.5.2.

| Arquitectura | HP | Learn rate | Capas | Acc | Acc_k^1 |
|--------------|---|------------|---------|-------|-----------|
| SNNTLBASIC | $\alpha = 5, \lambda = 0.01$ | 10^{-4} | FC, I5b | 0.041 | 19 |
| | $\alpha = 0.2, \lambda = 0.01$ | 10^{-4} | FC, I5b | 0.030 | 19 |
| | $\alpha = 20, \lambda = 0.01$ | 10^{-5} | FC, I5b | 0.035 | 20 |
| | $\alpha = 5, \lambda = 0.2$ | 10^{-4} | FC, I5b | 0.497 | 20 |
| | $\alpha = 10, \lambda = 0.2$ | 10^{-4} | FC, I5b | 0.500 | 21 |
| | $\alpha = 0.2, \lambda = 0.2$ | 10^{-4} | FC, I5b | 0.480 | 22 |
| SNNTLSEL | $\alpha = 20, \lambda = 0.01$ | 10^{-4} | FC, I5b | 0.032 | 21 |
| | $\alpha = 5, \lambda = 0.1$ | 10^{-6} | FC, I5b | 0.020 | 20 |
| | $\alpha = 20, \lambda = 0.1$ | 10^{-4} | FC, I5b | 0.024 | 20 |
| | $\alpha = 10, \lambda = 0.2$ | 10^{-4} | FC, I5b | 0.268 | 22 |
| | $\alpha = 5, \lambda = 0.2$ | 10^{-4} | FC, I5b | 0.279 | 22 |
| | $\alpha = 20, \lambda = 0.4$ | 10^{-4} | FC, I5b | 0.500 | 23 |
| SNNTLC | $\alpha = 5, \lambda = 0.2,$ $\alpha_{pen} = 0.2, \epsilon = 0.9$ | 10^{-4} | FC, I5b | 0.079 | 20 |
| | $\alpha = 5, \lambda = 0.2,$ $\alpha_{pen} = 0.5, \epsilon = 0.5$ | 10^{-4} | FC, I5b | 0.035 | 21 |
| | $\alpha = 5, \lambda = 0.2,$ $\alpha_{pen} = 0.2, \epsilon = 0.1$ | 10^{-4} | FC, I5b | 0.087 | 21 |
| | $\alpha = 0.2, \lambda = 0.01,$ $\alpha_{pen} = 0.1, \epsilon = 0.1$ | 10^{-6} | FC, I5b | 0.471 | 22 |
| | $\alpha = 10, \lambda = 0.2,$ $\alpha_{pen} = 0.5, \epsilon = 0.1$ | 10^{-6} | FC, I5b | 0.495 | 23 |
| | $\alpha = 5, \lambda = 0.2,$ $\alpha_{pen} = 0.1, \epsilon = 0.1$ | 10^{-6} | FC, I5b | 0.500 | 22 |

Continúa en la siguiente página.

Tabla 6.2 – Continuación de la anterior página.

| Arquitectura | HP | Learn rate | Capas | Acc | Acc_k^1 |
|--------------|---|------------|--------------|-------|-----------|
| SNNTLONLINE | $\alpha = 10, \lambda = 0.1,$ $T_B = B_A$ | 10^{-5} | FC, I5b | 0.033 | 23 |
| | $\alpha = 10, \lambda = 0.01,$ $T_B = B_H$ | 10^{-6} | FC, I5b | 0.491 | 24 |
| | $\alpha = 10, \lambda = 0.1,$ $T_B = B_A$ | 10^{-4} | FC, I5b | 0.033 | 24 |
| | $\alpha = 0.2, \lambda = 0.4,$ $T_B = B_H$ | 10^{-6} | FC, I5b | 0.397 | 25 |
| | $\alpha = 10, \lambda = 0.2,$ $T_B = B_H$ | 10^{-6} | FC, I5b | 0.429 | 26 |
| | $\alpha = 20, \lambda = 0.1,$ $T_B = B_A$ | 10^{-4} | FC, I5b | 0.491 | 24 |
| SNN SIGMOID | $R_D = 0.5, \lambda = 0$ | 10^{-4} | FC, I5b | 0.045 | 21 |
| | $R_D = 0.7, \lambda = 0.1$ | 10^{-5} | FC, I5b | 0.026 | 20 |
| | $R_D = 0.7, \lambda = 0.1$ | 10^{-4} | FC, I5b | 0.035 | 19 |
| | $R_D = 0.5, \lambda = 0$ | 10^{-4} | FC | 0.210 | 23 |
| | $R_D = 0.5, \lambda = 0$ | 10^{-5} | FC, I5b, I5a | 0.210 | 23 |
| | $R_D = 0.7, \lambda = 0.1$ | 10^{-3} | FC, I5b, I5a | 0.280 | 24 |

6.5.3. Evaluación de modelos en Validación Cruzada y discusión de resultados

En esta sección se van a analizar los resultados de CV. Los modelos seleccionados en hold-out, mostrados en la Tabla 6.2, fueron evaluados mediante CV 5-fold para tener una visión más fiable de su rendimiento. Las particiones en folds son exactamente las mismas en todos los experimentos.

Es importante que, a la hora de valorar los resultados obtenidos, se tenga en cuenta la poca cantidad de datos disponibles: 57 individuos positivos y 57 individuos negativos. Además, dado que se divide el conjunto de datos en entrenamiento, validación y test, los datos disponibles para entrenamiento son aún menores. Esto provoca que sea probable que los modelos estén fallando en los mismos casos debido a que sean especialmente complicados.

Al igual que en hold-out, de cada tipo de arquitectura se ha seleccionado el modelo que mejor tamaño de la muestra reducida obtuvo (menor Acc_K^1) y el modelo que mejor accuracy total (mayor Acc) logró. No obstante, para profundizar en el análisis de los resultados obtenidos, también se consideran los modelos que consiguieron mejor accuracy en test positivo (mayor Acc_{pos}) y mejor accuracy en test negativo (Acc_{neg}).

Mejor modelo en la reducción de la muestra de candidatos

La Tabla 6.3 recoge el mejor modelo de cada arquitectura en función del tamaño al que consiguen reducir la muestra de candidatos.

| Arquitectura | HP | Learn rate | Capas | Acc | Acc_k^1 | NFA |
|-------------------|---|------------|---------|--------|-----------|-----|
| SNNTLBASIC | $\alpha = 20, \lambda = 0.01$ | 10^{-5} | FC, I5b | 0.0533 | 12 | 10 |
| SNNTLSEL | $\alpha = 20, \lambda = 0.01$ | 10^{-4} | FC, I5b | 0.072 | 12 | 12 |
| SNNTLC | $\alpha = 0.2, \lambda = 0.01,$ $\alpha_{pen} = 0.1, \epsilon = 0.1$ | 10^{-6} | FC, I5b | 0.031 | 13 | 13 |
| SNNTLONLINE | $\alpha = 10, \lambda = 0.1,$ $T_B = B_A$ | 10^{-5} | FC, I5b | 0.100 | 13 | 13 |
| SNNSIGMOID | $R_D = 0.7, \lambda = 0.1$ | 10^{-4} | FC, I5b | 0.121 | 12 | 12 |

Tabla 6.3: Resultados de CV: se recoge el mejor modelo de cada arquitectura según el tamaño de la muestra reducida (menor Acc_k^1). La Tabla muestra la combinación de hiperparámetros del modelo, HP, el *learning rate* del optimizador del entrenamiento, las capas de *FaceNet* reentrenadas para cada modelo, el *accuracy* total (*Acc*) y el máximo número de imágenes faciales incorrectamente aceptadas en test negativo (*NFA*). El significado de los hiperparámetros de los modelos puede comprobarse en la Sección 5.2.4 y el de las capas de *FaceNet* en la Sección 6.5.2.

Los modelos que optimizan la Acc_k^1 tienden a tener peor rendimiento en la *accuracy* total *Acc* (en comparación a otros modelos que no mejoran tanto la Acc_k^1). Esto puede ser debido a que estos modelos se especialicen en minimizar el número de clases parecidas, esto es, el número de individuos parecidos al individuo de test. Como se puede ver en la Figura 6.16, el número de predicciones correctas es muy bajo (inferior al 15 %) para los tres modelos que obtienen mejor Acc_k^1 (destacados en la Tabla 6.3), mientras que hay un rápido incremento en el *top-k accuracy*. El aumento del *top-k accuracy* es un indicador de que las imágenes faciales correctas son predichas entre las más semejantes de la muestra, aunque no sean las devueltas en la identificación. El comportamiento de estos modelos en test negativo es similar: las predicciones correctas son muy escasas, pero el número de imágenes faciales incorrectamente aceptadas es pequeño (ya que la curva NFA aumenta rápidamente, como se puede ver en la Figura 6.17).

En conjunto, estos tres modelos anteriores consiguen reducir la muestra de candidatos al 12 % de la original (en test positivo), mientras que en test negativo destaca el modelo SNNTLBASIC ya que nunca acepta de forma incorrecta más de 10 imágenes (el NFA es 10). La importancia de este aspecto es grande: al recibir cualquier tipo de imagen craneal desconocida (tanto si fuese positiva como negativa) se establece con total certeza que entre las 12 imágenes faciales más semejantes estará la real si se dispone de ella. Para aclarar este punto, esto no sería posible si el modelo ofreciese un $Acc_k^1 = 12$

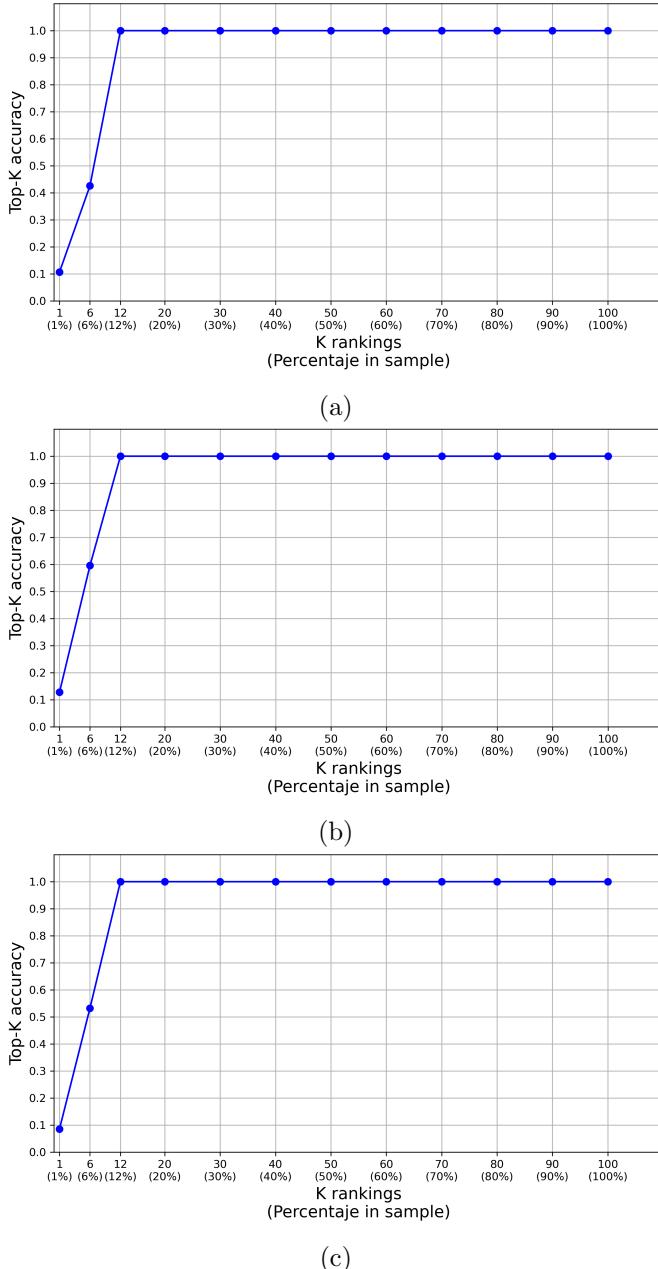


Figura 6.16: Curvas CMC de los modelos con mayor reducción de la muestra de candidatos (menor Acc_k^1 , destacados en la Tabla 6.3). De arriba a abajo, SNNTLBASIC (6.16a), SNNTLSEL (6.16b) y SNNSIGMOID (6.16c): los tres consiguen reducir la muestra al 12 % de la original.

pero un $NFA = 20$, ya que para una imagen craneal negativa se podría llegar a retornar incorrectamente hasta 20 imágenes.

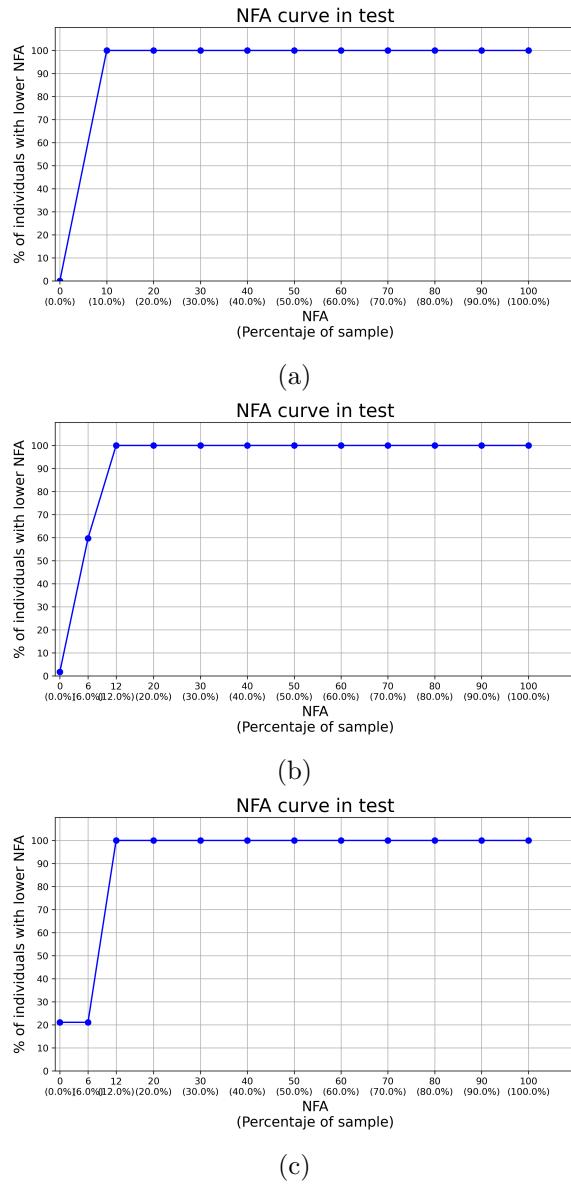


Figura 6.17: Curvas NFA de los modelos con mayor reducción de la muestra de candidatos (menor Acc_k^1 , en destacados en la Tabla 6.3). De arriba a abajo, SNNTLBASIC (6.17a), SNNTLSEL (6.17b) y SNNSIGMOID (6.17c): el primero acepta, como mucho, 10 imágenes faciales de forma incorrecta y el segundo y tercero 12.

Mejor modelo según el accuracy conseguido

La Tabla 6.4 recoge el mejor modelo de cada arquitectura en función del *accuracy* total (en test positivo y negativo) conseguido.

| Arquitectura | HP | Learn rate | Capas | Acc | Acc _{pos} | Acc _{neg} | Acc _k ¹ |
|-------------------|--|------------|---------|--------------|--------------------|--------------------|-------------------------------|
| SNNTLBASIC | $\alpha = 0.2, \lambda = 0.2$ | 10^{-4} | FC, I5b | 0.400 | 0.000 | 0.800 | 15 |
| SNNTLSEL | $\alpha = 10, \lambda = 0.2$ | 10^{-4} | FC, I5b | 0.318 | 0.042 | 0.593 | 17 |
| SNNTLC | $\alpha = 5, \lambda = 0.2,$ $\alpha_{pen} = 0.2, \epsilon = 0.9$ | 10^{-4} | FC, I5b | 0.331 | 0.064 | 0.597 | 16 |
| SNNTLONLINE | $\alpha = 0.2, \lambda = 0.4,$ $T_B = B_H$ | 10^{-6} | FC, I5b | 0.366 | 0.022 | 0.710 | 15 |
| SNNSIGMOID | $R_D = 0.7, \lambda = 0.1$ | 10^{-5} | FC, I5b | 0.396 | 0.064 | 0.727 | 30 |

Tabla 6.4: Resultados de CV: se recoge el mejor modelo de cada arquitectura según el *accuracy* total, en test positivo y negativo, (mayor *Acc*). La Tabla muestra la combinación de hiperparámetros del modelo, HP, el *learning rate* del optimizador del entrenamiento, las capas de *FaceNet* reentrenadas para cada modelo y el tamaño de la muestra reducida (*Acc*_{*k*}¹). El significado de los hiperparámetros de los modelos puede comprobarse en la Sección 5.2.4 y el de las capas de *FaceNet* en la Sección 6.5.2.

En la misma línea de lo que ocurría con los modelos que mejor reducción de la muestra de candidatos conseguían, los modelos que optimizan la *accuracy* total tienden a tener peores *Acc*_{*k*}¹. Este hecho refuerza la hipótesis de que los modelos se especializan en una tarea o en la otra: o bien reducen la muestra considerablemente o bien aumentan el número de predicciones correctas. La Figura 6.18 visualiza el *trade-off* del modelo con mejor *accuracy* (destacado en la Tabla 6.4): el modelo realiza un 80 % de las predicciones negativas correctamente, mientras que las positivas no son tan buenas y el *Acc*_{*k*}¹ empeora visiblemente.

En todos los resultados presentados hasta ahora, el *accuracy* se corresponde con el promedio de *accuracy* en test positivo y test negativo. No obstante, como en este TFG se ha trabajado de forma separada con estos dos conjuntos (dada su diferente naturaleza⁴), resulta interesante también analizar en paralelo el comportamiento de los modelos en ambos conjuntos de test. Las Tablas 6.5 y 6.6 recogen los resultados de las diferentes arquitecturas en test positivo y test negativo, respectivamente.

Un hecho que se comprueba en todos los casos es que el número de predicciones correctas en Test Positivo es notablemente inferior al número de predicciones correctas en Test Negativo. El modelo SNNTLSEL es el que consigue realizar el mayor número de predicciones positivas de forma correcta, un 12.7 % de ellas (ver Tabla 6.5), mientras que el modelo SNNTLBASIC es el que mejores predicciones negativas realiza, ya que un 72.7 % de ellas son correctas (ver Tabla 6.6). El hecho de presentar un menor rendimiento

⁴Recordar que el conjunto de test positivo contiene imágenes craneales positivas, cuyas imágenes faciales asociadas están en la BD de imágenes faciales de test, mientras que el de test negativo contiene imágenes craneales negativas, para las que no se dispone de una imagen facial asociada.

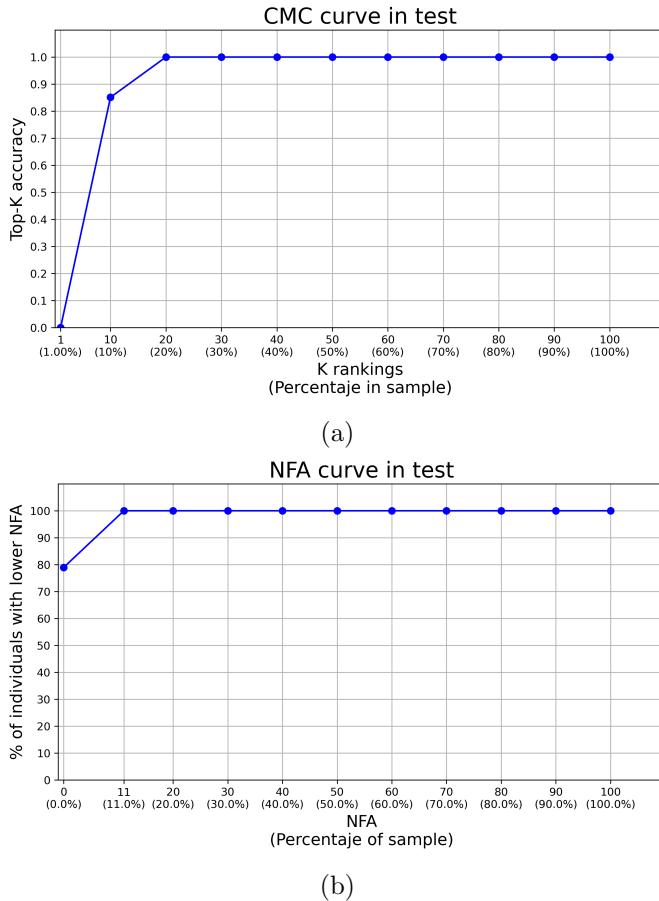


Figura 6.18: Curva CMC (6.18a) y NFA (6.18b) del modelo que consigue mayor *accuracy*. Destaca, sobre todo, la calidad de las predicciones en test negativo.

en la identificación positiva pues ser debido a que, dado el escaso número de imágenes positivas, haya imágenes faciales complicadas que se asemejan a mucho a imágenes craneales positivas, provocando que los modelos fallen en estas.

Análisis adicionales

Es interesante destacar un aspecto hallado en la experimentación, que causa cierta sorpresa: se ha comprobado que el modelo SNNTLBASIC se encuentra entre los tres mejores modelos de Acc_k^1 y es el mejor modelo de Acc . Este modelo implementa la Estrategia de generación offline de triplets sin ninguna mejora (ni selección de triplets complicadas ni TLC), lo cual contrasta notablemente con los estudios previos en el ámbito de reconocimiento facial mediante SNN [105, 108]. Esta estrategia de generación de triplets

| Arquitectura | HP | Learn rate | Capas | Acc_{pos} |
|-----------------|--|------------|---------|--------------|
| SNNTLBASIC | $\alpha = 20, \lambda = 0.01$ | 10^{-5} | FC, I5b | 0.107 |
| SNNTLSEL | $\alpha = 20, \lambda = 0.01$ | 10^{-4} | FC, I5b | 0.127 |
| SNNTLC | $\alpha = 5, \lambda = 0.2,$ $\alpha_{pen} = 0.1, \epsilon = 0.1$ | 10^{-6} | FC, I5b | 0.084 |
| SNNTLONLINE | $\alpha = 20, \lambda = 0.1,$ $T_B = B_A$ | 10^{-4} | FC, I5b | 0.067 |
| SNNSIGMOID | $R_D = 0.7, \lambda = 0.1$ | 10^{-5} | FC, I5b | 0.107 |

Tabla 6.5: Resultados de CV: se recoge el mejor modelo de cada arquitectura según el *accuracy* en test positivo (mayor Acc_{pos}). Se destaca en rojo el mejor modelo. La Tabla muestra la combinación de hiperparámetros del modelo, HP, el *learning rate* del optimizador del entrenamiento, las capas de *FaceNet* reentrenadas para cada modelo. El significado de los hiperparámetros de los modelos puede comprobarse en la Sección 5.2.4 y el de las capas de *FaceNet* en la Sección 6.5.2.

| Arquitectura | HP | Learn rate | Capas | Acc_{neg} |
|-------------------|--|------------|---------|--------------|
| SNNTLBASIC | $\alpha = 0.2, \lambda = 0.2$ | 10^{-4} | FC, I5b | 0.800 |
| SNNTLSEL | $\alpha = 20, \lambda = 0.4$ | 10^{-4} | FC, I5b | 0.600 |
| SNNTLC | $\alpha = 5, \lambda = 0.2,$ $\alpha_{pen} = 0.2, \epsilon = 0.1$ | 10^{-4} | FC, I5b | 0.599 |
| SNNTLONLINE | $\alpha = 0.2, \lambda = 0.4,$ $T_B = B_H$ | 10^{-6} | FC, I5b | 0.711 |
| SNNSIGMOID | $R_D = 0.7, \lambda = 0.1$ | 10^{-5} | FC, I5b | 0.727 |

Tabla 6.6: Resultados de CV: se recoge el mejor modelo de cada arquitectura según el *accuracy* en test negativo (mayor Acc_{neg}). La Tabla muestra la combinación de hiperparámetros del modelo, HP, el *learning rate* del optimizador del entrenamiento, las capas de *FaceNet* reentrenadas para cada modelo. El significado de los hiperparámetros de los modelos puede comprobarse en la Sección 5.2.4 y el de las capas de *FaceNet* en la Sección 6.5.2.

podría ser más eficaz en este problema, en comparación con reconocimiento facial, por lo siguiente:

- Es posible que las tripletas aleatorias sean más complicadas en el problema tratado en este TFG que en reconocimiento facial, dada la naturaleza de ambos problemas.
- Al disponer de una menor cantidad de datos en este trabajo, frente a los estudios del estado del arte de reconocimiento facial (que utilizan millones de imágenes faciales diferentes), el número de tripletas generadas es menor y, por tanto, el número de tripletas sencillas es menor.

Esto podría significar que el entrenamiento se vea menos perjudicado que en tareas de reconocimiento facial por la presencia de esas tripletas sencillas.

Otro detalle importante a destacar en cuanto a las propuestas realizadas en este TFG es en relación a las diferentes estrategias de selección del umbral de decisión. Se ha comprobado experimentalmente que el umbral de decisión obtenido mediante las tres estrategias es el mismo, ya que tiende a converger en el mismo punto (ver Figura 6.19). Por esta razón, no se han tenido en cuenta en las tablas de resultados, ya que han sido los mismos para los tres umbrales.

Finalmente, destacar la importancia de la elección del umbral α y del regulador de la regularización L2 en el entrenamiento con TL. Gracias a la batería experimental realizada con hold-out, se ha comprobado que una mala elección provoca que el modelo colapse y no aprenda nada, como se puede ver en la Figura 6.20. Al colapsar, el modelo proyecta todas las entradas al mismo punto del espacio de *embeddings* (o a puntos muy cercanos) y deja de ser capaz de aprender.

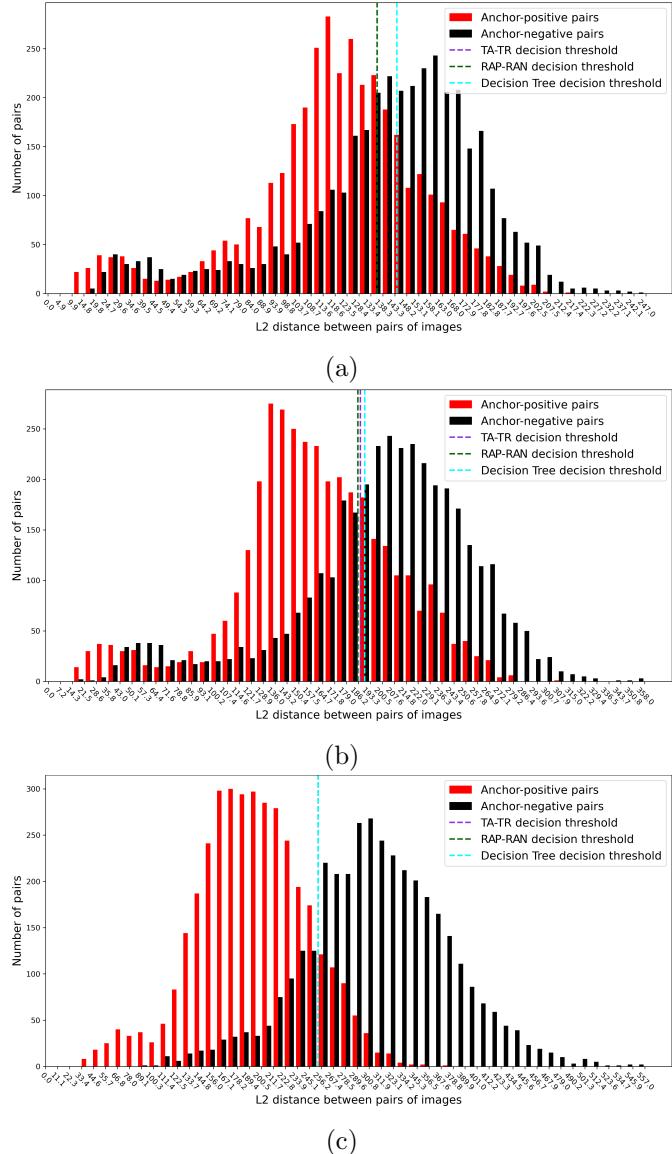


Figura 6.19: Evolución de los umbrales de decisión obtenidos con los tres métodos propuestos a medida que el modelo de DL aprende a separar las distancias entre pares positivos (en rojo) y pares negativos (en negro). Al poco de comenzar el entrenamiento, los umbrales aparecen claramente diferenciados (6.19a), mientras que a medida que los pares positivos y negativos se distancian los umbrales van convergiendo a un punto cercano (6.19b), terminando en el mismo punto (6.19c). Estos ejemplos han sido generados utilizando una SNN entrenada con TL, donde las entradas son tripletas.

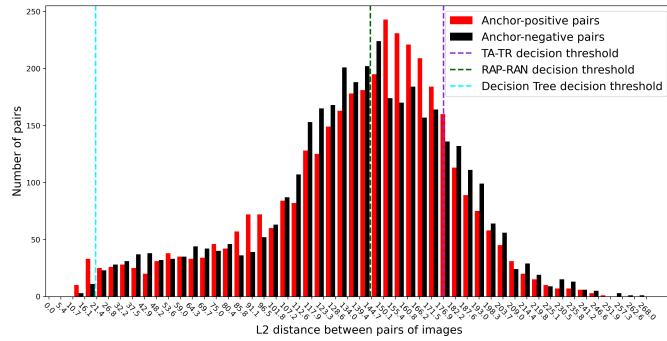


Figura 6.20: Ejemplo de modelo de SNN entrenada con TL colapsado durante el entrenamiento. En este caso, se entrenó una SNNTLBASIC con $\alpha = 0.2$ con $\lambda = 0.001$: dado este umbral tan pequeño y esta escasa regularización, la mayoría de pares de imágenes se consideran correctamente clasificadas en entrenamiento por lo que la red no aprende a separarlas realmente.

Capítulo 7

Conclusiones y trabajos futuros

La automatización de la Identificación Humana (IDH) es un problema complejo y difícil de resolver. Se trata de un campo cuya investigación ha comenzado recientemente y en el que los conjuntos de datos disponibles no son muy amplios, ya que su adquisición es complicada. En este TFG, se ha realizado un minucioso estudio del arte, que se ha extendido con el objetivo de analizar las corrientes más modernas en la solución de problemas similares. Se partió de un modelo de Deep Learning (DL) para reconocimiento facial, *FaceNet* [105] y se adaptó a la problemática de la IDH craneofacial. Dicha casuística se ocupa de la identificación a partir de imágenes 2D craneales y faciales. Su complejidad es particularmente importante dados los escasos estudios en el campo y la limitada cantidad disponible de imágenes craneales emparejadas con su correspondiente imagen facial.

Este trabajo **ha sido el primero en proponer una solución basada en Deep Learning a este problema**, proponiendo cinco modelos diferentes con el objetivo de explorar la aplicación de estas técnicas a este problema. Las arquitecturas propuestas **han conseguido reducir la muestra de candidatos en un 88 % de la original**. Es decir, se ha logrado reducir la muestra de candidatos a un 12 % de su tamaño original. Estos resultados son especialmente destacables dada la escasa disponibilidad de datos y la gran dependencia de conjuntos de datos extensos que presentan las técnicas de DL. No obstante, los modelos presentados tienen más dificultades a la hora de clasificar correctamente las imágenes craneales cuya imagen facial se encuentra en la base de datos de referencia (imágenes craneales positivas). En concreto, se consigue emparejar de forma exitosa un 12.7 % imágenes craneales positivas con su respectiva imagen facial de entre todas las candidatas. Por todo lo anterior, **es viable el empleo del sistema automatizado propuesto en este trabajo en el escenario de la IDH para la reducción de candidatos**.

Si se repasan los objetivos de este trabajo, se puede comprobar que se han cumplido todos: se han analizado detenidamente las técnicas aplicadas a este problema y similares, se ha explorado el conjunto de modelos 3D disponible y se ha diseñado una herramienta de generación de imágenes 2D craneales a partir de estos, se han considerado la diferentes técnicas de DL aplicables al problema en cuestión, planteando varios modelos que permitan automatizar el proceso de la IDH y, finalmente, se han evaluado las diferentes propuestas para analizar su validez para el problema en cuestión. Para la consecución de los anteriores puntos ha sido necesario aunar conocimientos de diversos campos (Antropología Forense, Deep Learning, Visión por Computador), siendo una tarea de integración de múltiples técnicas especialmente complicada por la complejidad del problema afrontado. Dada la naturaleza investigadora de este TFG y el interés por el desarrollo futuro de nuevos modelos aplicables a este mismo problema, se ha liberado en GitHub¹ el código desarrollado en este trabajo para favorecer nuevos estudios en la materia.

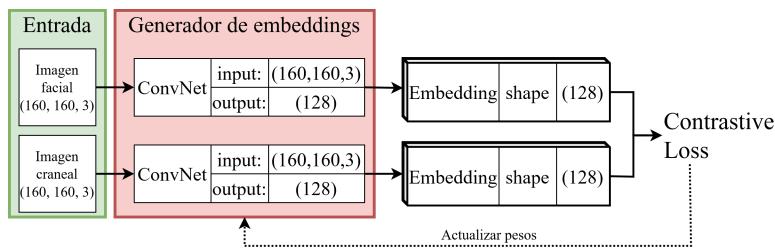


Figura 7.1: Estructura de SNN entrenada con Contrastive Loss propuesta como trabajo futuro. Elaboración propia.

Durante el desarrollo de este TFG se ha constatado que las técnicas de DL pueden tener una aplicación interesante en la IDH con imágenes faciales. El futuro es esperanzador y, por ello, se presentan varias líneas de trabajo en este ámbito de investigación. En primer lugar, en cuanto a las técnicas de DL empleadas, se propone la búsqueda de una nueva estrategia de generación de tripletas en las SNN entrenadas con *Triplet Loss* que permita aumentar la calidad de las tripletas de entrenamiento. También se plantea el empleo de SNN entrenadas con *Contrastive Loss* [80] (ver Figura 7.1): esta función de pérdida podría permitir que los modelos aprendan una mejor distinción entre las imágenes faciales positivas y las negativas, pudiendo ser este un punto importante para mejorar el rendimiento de los modelos. Así mismo, otra alternativa interesante es la aplicación de CNN que reciban como entrada un bloque con dos imágenes concatenadas e identifique si pertenecen a la misma persona o no (ver Figura 7.2).

En segundo lugar, se propone el estudio de la integración del dataset

¹https://github.com/MarioVillar/TFG_IDH

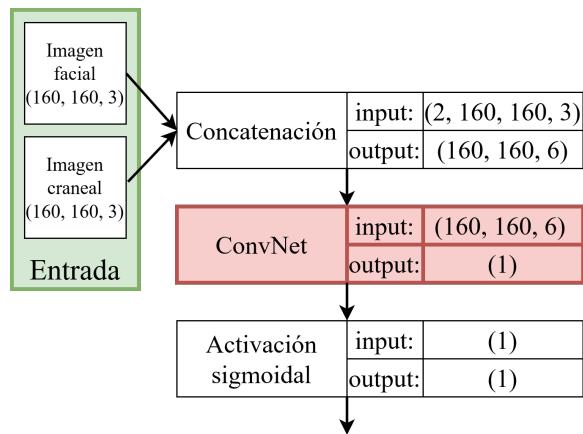


Figura 7.2: Estructura de CNN que recibe dos imágenes concatenadas en un único volumen, propuesta como trabajo futuro. Elaboración propia.

IdentifyMe², recientemente hecho público, con el dataset empleado en este trabajo. Y, finalmente, se plantea el estudio de la ampliación del *data augmentation* realizado sobre las imágenes faciales utilizadas en el entrenamiento de los modelos de DL para aumentar la varianza de los datos.

²<http://iab-rubric.org/index.php/identify>

Bibliografía

- [1] F. Abdu, Y. Zhang, and Z. Deng. Activity classification based on feature fusion of FMCW radar human motion micro-doppler signatures. *IEEE Sensors Journal*, 22(9):8648–8662, 2022.
- [2] T. Abdullah, Y. Bazi, M. Al Rahhal, M. Mekhalfi, L. Rangarajan, and M. Zuair. Textrs: Deep bidirectional triplet network for matching text to remote sensing images. *Remote Sensing*, 12(3):405, 2020.
- [3] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin. *Learning From Data*. AMLBook, 2012.
- [4] B. Adams. *Forensic Anthropology*. Chelsea House Publications, 2006.
- [5] I. Adjabi, A. Ouahabi, A. Benzaoui, and A. Taleb-Ahmed. Past, present, and future of face recognition: A review. *Electronics*, 9(8):1188, 2020.
- [6] W. Al Noumah, A. Jafar, and K. Al Joumaa. Using parallel pre-trained types of dcnn model to predict breast cancer with color normalization. *BMC Research Notes*, 15(1):14, 2022.
- [7] J. E. Allanson, C. Cunniff, H. E. Hoyne, J. McGaughran, M. Muenke, and G. Neri. Elements of morphology: Standard terminology for the head and face. *American Journal of Medical Genetics Part A (AJMG)*, 149A(1):6–28, 2009.
- [8] E. Alpaydin. *Introduction to Machine Learning*. MIT Press, 2009.
- [9] F. Andaló and S. Goldenstein. Computer vision methods applicable to forensic science. In *Workshop of Theses and Dissertations, Conference on Graphics, Patterns and Images (SIBGRAPI)*, 2013.
- [10] Asociación Educar para el Desarrollo Humano. Ilustración neurociencias: Cráneo. <https://asociacioneducar.com/craneo>, 2019.
- [11] B. Baheti, S. Innani, S. Gajre, and S. Talbar. Eff-unet: A novel architecture for semantic segmentation in unstructured environment. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR)*, volume 2020-June, pages 1473–1481, 2020.
- [12] H. Basak, R. Kundu, P. Singh, M. Ijaz, M. Woźniak, and R. Sarkar. A union of deep learning and swarm-based optimization for 3d human action recognition. *Scientific Reports*, 12(1):5494, 2022.
- [13] Y. Bazi, M. Rahhal, H. Alhichri, and N. Alajlan. Simple yet effective fine-tuning of deep cnns using an auxiliary classification loss for remote sensing scene classification. *Remote Sensing*, 11(24):2908, 2019.

- [14] E. Bermejo, C. Campomanes-Álvarez, A. Valsecchi, O. Ibáñez, S. Damas, and O. Cordón. Genetic algorithms for skull-face overlay including mandible articulation. *Information Sciences*, 420:200–217, 2017.
- [15] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
- [16] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
- [17] W. Bledsoe. The model method in facial recognition. Technical report, Panoramic Research, Inc., 1964.
- [18] B. Campomanes Álvarez, O. Ibáñez, F. Navarro, I. Aleman, M. Boteilla, S. Damas, and O. Cordón. Computer vision and soft computing for automatic skull-face overlay in craniofacial superimposition. *Forensic Science International (FSI)*, 245:77–86, 2014.
- [19] G. Chechik, V. Sharma, U. Shalit, and S. Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research (JMLR)*, 11:1109–1135, 2010.
- [20] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun. Bayesian face revisited: A joint formulation. In *European Conference on Computer Vision (ECCV)*, pages 566–579, 2012.
- [21] Y. Cheng, J. Zhao, Z. Wang, Y. Xu, K. Jayashree, S. Shen, and J. Feng. Know you at one glance: A compact vector representation for low-shot learning. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1924–1932, 2017.
- [22] F. Chollet. *Deep Learning with Python*. Manning Publications Co., 2018.
- [23] A. Christensen, N. Passalacqua, and E. Bartelink. *Forensic Anthropology: Current Methods and Practice*. Academic Press, 2019.
- [24] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. Meshlab: an open-source mesh processing tool. In *Eurographics Italian Chapter Conference*, 2008.
- [25] S. Dahal and N. Kumar Agrawal. Forensic human identification methods: A review. *International Journal Of Scientific Research (IJSR)*, 6(11):26, 2017.
- [26] S. Damas, O. Cordón, and O. Ibáñez. *Handbook on Craniofacial Superimposition: The MEPROCS Project*. Springer International Publishing, 2020.

- [27] H. H. de Boer, Z. Obertová, E. Cunha, P. Adalian, E. Baccino, T. Fracasso, E. Kranioti, P. Lefévre, N. Lynnerup, A. Petaros, A. Ross, M. Steyn, and C. Cattaneo. Strengthening the role of forensic anthropology in personal identification: Position statement by the board of the forensic anthropology society of europe (fase). *Forensic Science International (FSI)*, 315:110456, 2020.
- [28] J. Deng, S. Cheng, N. Xue, Z. Y., and S. Zafeiriou. Uv-gan: Adversarial facial uv map completion for pose-invariant face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7093–7102, 2018.
- [29] D. Dewangan and S. Sahu. Optimized convolutional neural network for road detection with structured contour and spatial information for intelligent vehicle system. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 36(6):2252002, 2022.
- [30] A. Dhillon and G. Verma. Convolutional neural network: a review of models, methodologies and applications to object detection. *Progress in Artificial Intelligence*, 9(2):85–112, 2019.
- [31] S. Dogan, P. Datta Barua, H. Kutlu, M. Baygin, H. Fujita, T. Tuncer, and U. Acharya. Automated accurate fire detection system using ensemble pretrained residual network. *Expert Systems with Applications (ESA)*, 203:117407, 2022.
- [32] S. Du. Understanding deep self-attention mechanism in convolution neural networks. <https://bit.ly/3I8DGz5>, 2020.
- [33] E.-S. El-Dahshan, M. Bassiouni, A. Hagag, R. Chakrabortty, H. Loh, and U. Acharya. Rescovidtcnnnet: A residual neural network-based framework for covid-19 detection using tcn and ewt with chest x-ray images. *Expert Systems with Applications (ESA)*, 204:117410, 2022.
- [34] A. Esteva, K. Chou, S. Yeung, N. Naik, A. Madani, A. Mottaghi, Y. Liu, E. Topol, J. Dean, and R. Socher. Deep learning-enabled medical computer vision. *npj Digital Medicine*, 4(1):5, 2021.
- [35] A. Esteva, B. Kuprel, R. Novoa, J. Ko, S. Swetter, H. Blau, and S. Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115–118, 2017.
- [36] W. D. Ferreira, C. B. Ferreira, G. da Cruz Júnior, and F. Soares. A review of digital image forensics. *Computers & Electrical Engineering*, 85:106685, 2020.

- [37] P. Gader and M. Khabou. Automatic feature generation for handwritten digit recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 18(12):1256–1261, 1996.
- [38] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.
- [39] Google Inc. Google colaboratory. <https://colab.research.google.com/>, 2021.
- [40] J. Gowrishankar, T. Narmadha, R. Mohan, and Y. Natarajan. Convolutional neural network classification on 2d craniofacial images. *International Journal of Grid and Distributed Computing (IJGDC)*, 13:1026–1032, 2020.
- [41] D. Gray, S. Brennan, and H. Tao. Evaluating appearance models for recognition, reacquisition, and tracking. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, 2007.
- [42] J. L. Gutiérrez, M. B. Valverde, M. M. Hernández, J. P. Bonilla, F. J. G. Herrero, T. C. del Rey, F. J. R. Sancho, I. P. Cachinero, and A. C. G. Murillas. *Informe anual de personas desparecidas durante el año 2021*. Ministerio de Interior, Gobierno del Reino de España, 2022.
- [43] H. He, M. Chen, T. Chen, and D. Li. Matching of remote sensing images with complex background variations via siamese convolutional neural network. *Remote Sensing*, 10(2):355, 2018.
- [44] H. He, M. Chen, T. Chen, D. Li, and P. Cheng. Learning to match multitemporal optical satellite images using multi-support-patches siamese networks. *Remote Sensing Letters (RSL)*, 10(6):516–525, 2019.
- [45] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [46] Z. He. Deep learning in image classification: A survey report. In *International Conference on Information Technology and Computer Application (ITCA)*, pages 174–177, 2020.
- [47] A. Hermans, L. Beyer, and B. Leibe. In defense of the triplet loss for person re-identification, 2017.
- [48] W. Hongdan, S. SalmiJamali, C. Zhengping, S. Qiaojuan, and R. Le. An intelligent music genre analysis using feature extraction and classification using deep learning techniques. *Computers and Electrical Engineering*, 100:107978, 2022.

- [49] M. Houck and J. Siegel. *Fundamentals of Forensic Science*. Academic Press, 2006.
- [50] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu. Squeeze-and-excitation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 42:2011–2023, 2020.
- [51] Human Skull. <https://bit.ly/3PNGgx0>.
- [52] O. Ibáñez, O. Cordón, S. Damas, and J. Santamaría. An advanced scatter search design for skull-face overlay in craniofacial superimposition. *Expert Systems with Applications (ESA)*, 39(1):1459–1473, 2012.
- [53] O. Ibáñez, A. Valsecchi, F. Cavalli, M. I. Huete, B. R. Campomanes-Alvarez, C. Campomanes-Alvarez, R. Vicente, D. Navega, A. Ross, C. Wilkinson, R. Jankauskas, K. Imaizumi, R. Hardiman, P. T. Jayaprakash, E. Ruiz, F. Molinero, P. Lestón, E. Veselovskaya, A. Abramov, M. Steyn, J. Cardoso, D. Humpire, L. Lusnig, D. Gibelli, D. Mazzarelli, D. Gaudio, F. Collini, and S. Damas. Study on the criteria for assessing skull-face correspondence in craniofacial superimposition. *Legal Medicine*, 23:59–70, 2016.
- [54] International Committee of the Red Cross. Missing persons and international humanitarian law. <https://www.icrc.org/en/document/protected-persons/missing-persons>, 2010.
- [55] S. Jadon. An overview of deep learning architectures in few-shot learning domain, 2020.
- [56] S. Jadon and A. Garg. *Hands-On One-shot Learning with Python*. Packt Publishing, 2020.
- [57] G. Jayalakshmi, H. Khalaf, A. Farhadi, S. Al-Barzinji, S. Mahmood, S.-D. Najim, M. Hutaihit, S. Nejrs, R. Mahdawi, and A. Abdulbaqi. Detection of covid-19 from radiology modalities and identification of prognosis patterns. *International Journal of Nonlinear Analysis and Applications (IJNAA)*, 13(1):1351–1365, 2022.
- [58] L. Jiao and J. Zhao. A survey on the new generation of deep learning in image processing. *IEEE Access*, 7:172231–172263, 2019.
- [59] A. Jose, S. Yan, and I. Heisterklaus. Binary hashing using siamese neural networks. In *IEEE International Conference on Image Processing (ICIP)*, pages 2916–2920, 2017.

- [60] S. Kadry, V. Rajinikanth, D. Taniar, R. Damaševičius, and X. Valencia. Automated segmentation of leukocyte from hematological images—a study using various cnn schemes. *Journal of Supercomputing*, 78(5):6974–6994, 2022.
- [61] M. Kan, S. Shan, H. Chang, and X. Chen. Stacked progressive auto-encoders (spae) for face recognition across poses. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1883–1890, 2014.
- [62] A. Khan, A. Sohail, U. Zahoor, and A. S. Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 53(8):5455–5516, 2020.
- [63] H. Kim, A. Cosa-Linan, N. Santhanam, M. Jannesari, M. Maros, and T. Ganslandt. Transfer learning for medical image classification: a literature review. *BMC Medical Imaging*, 22(1):69, 2022.
- [64] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.
- [65] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 2, pages 1097–1105, 2012.
- [66] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.
- [67] C. Kwan, M. S. Kang, S. G. Nuara, J. C. Gourdon, D. Bédard, C. L. Tardif, R. Hopewell, K. Ross, H. Bdair, A. Hamadjida, G. Massarweh, J.-P. Soucy, W. Luo, E. del Cid Pellitero, I. Shlaifer, T. M. Durcan, E. A. Fon, P. Rosa-Neto, S. Frey, and P. Huot. Co-registration of imaging modalities (mri, ct and pet) to perform frameless stereotaxic robotic injections in the common marmoset. *Neuroscience*, 480:143–154, 2022.
- [68] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
- [69] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.
- [70] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

- [71] F.-F. Li, J. Johnson, and S. Yeung. Lecture 5: Convolutional neural networks. http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture05.pdf, 2018.
- [72] G. Litjens, T. Kooi, B. Bejnordi, A. Setio, F. Ciompi, M. Ghafoorian, J. van der Laak, B. van Ginneken, and C. Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42:60–88, 2017.
- [73] L. Liu, J. Chen, P. W. Fieguth, G. Zhao, R. Chellappa, and M. Pietikäinen. From bow to cnn: Two decades of texture representation for texture classification. *International Journal of Computer Vision (IJCV)*, 127:74–109, 2018.
- [74] L. Liu, W. Ouyang, X. Wang, P. W. Fieguth, J. Chen, X. Liu, and M. Pietikäinen. Deep learning for generic object detection: A survey. *International Journal of Computer Vision (IJCV)*, 128:261–318, 2019.
- [75] Z. Liu, X. Xiao, C. Li, S. Ma, and D. Rangyu. Optimizing convolutional neural networks on multi-core vector accelerator. *Parallel Computing*, 112:102945, 2022.
- [76] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 07-12-June-2015, pages 431–440, 2015.
- [77] Loyola University Chicago Stritch School of Medicine. Utility of imaging modalities in neurological disorders. https://www.stritch.luc.edu/lumen/MedEd/Radio/curriculum/Neuroscience/Image_Modalities_2013.htm.
- [78] F. Marturana and S. Tacconi. A machine learning-based triage methodology for automated categorization of digital media. *Digital Investigation*, 10:193–204, 2013.
- [79] V. Mazzia, F. Salvetti, and M. Chiaberge. Efficient-capsnet: capsule network with self-attention routing. *Scientific Reports*, 11(1):14634, 2021.
- [80] I. Melekhov, J. Kannala, and E. Rahtu. Siamese network features for image matching. In *Proceedings - International Conference on Pattern Recognition (ICPR)*, pages 378–383, 2016.
- [81] P. Mesejo, R. Martos, Ó. Ibáñez, J. Novo, and M. Ortega. A survey on artificial intelligence techniques for biomedical image analysis in skeleton-based forensic human identification. *Applied Sciences*, 10(14):4703, 2020.

- [82] A. Mishra, P. Gupta, and P. Tewari. Global u-net with amalgamation of inception model and improved kernel variation for mri brain image segmentation. *Multimedia Tools and Applications*, 81(16):23339–23354, 2022.
- [83] O. Moindrot. Triplet loss and online triplet mining in tensorflow. <https://omoindrot.github.io/triplet-loss>, 2018.
- [84] H. Moses Daluz. *Fundamentals of Fingerprint Analysis*. CRC Press, 2021.
- [85] R. Mukundan. *3D Mesh Processing and Character Animation*. Springer Nature, 2022.
- [86] Médicos sin Fronteras. Desastres naturales. <https://www.msf.es/nuestra-accion/desastres-naturales>.
- [87] S. N. Byers. *Introduction to Forensic Anthropology*. Routledge, 2016.
- [88] S. Nagpal, M. Singh, A. Jain, R. Singh, M. Vatsa, and A. Noore. On matching skulls to digital face images: A preliminary approach. In *IEEE International Joint Conference on Biometrics (IJCB)*, pages 813–819, 2017.
- [89] H. Noor, A. Decker, C. Gibson-McDonald, S. C. Arely, U.-T. Baik, and M. Klinkne. *Global report on missing people*. International Commission on Missing Persons, 2021.
- [90] G. Porter and G. Doran. An anatomical and photographic technique for forensic facial identification. *Forensic Science International (FSI)*, 114:97–105, 2000.
- [91] R. Pressman. *Ingeniería del software*. McGraw Hill, 2010.
- [92] E. U. H. Qazi, T. Zia, and A. Almorjan. Deep learning-based digital image forgery detection system. *Applied Sciences*, 12(6):2851, 2022.
- [93] X. Qi and L. Zhang. Face recognition via centralized coordinate learning, 2018.
- [94] R. R. Skeleton. *A Survey of the Forensic Sciences*. Lulu Press, Inc., 2010.
- [95] E. Raczkó, M. Krówczyńska, and E. Wilk. Asbestos roofing recognition by use of convolutional neural networks and high-resolution aerial imagery. testing different scenarios. *Building and Environment*, 217:109092, 2022.

- [96] R. Ranjan, C. D. Castillo, and R. Chellappa. L2-constrained softmax loss for discriminative face verification, 2017.
- [97] R. Ranjan, S. Sankaranarayanan, A. Bansal, N. Bodla, J. C. Chen, V. M. Patel, C. D. Castillo, and R. Chellappa. Deep learning for understanding faces: Machines may be just as good, or better, than humans. *IEEE Signal Processing Magazine (SPM)*, 35:66–83, 2018.
- [98] S. Ravichandiran. *Hands-On Deep Learning Algorithms with Python*. Packt Publishing, 2019.
- [99] E. Richardson, M. Sela, R. Or-El, and R. Kimmel. Learning detailed face reconstruction from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5553–5562, 2017.
- [100] H. Ritchie and M. Roser. Natural disasters. *Our World in Data*, 2014.
- [101] M. Rombolotti, F. Sangalli, D. Cerullo, A. Remuzzi, and E. Lanzarone. Automatic cyst and kidney segmentation in autosomal dominant polycystic kidney disease: Comparison of u-net based methods. *Computers in Biology and Medicine*, 146:105431, 2022.
- [102] S. Roy, M. Harandi, R. Nock, and R. Hartley. Siamese networks: The tale of two manifolds. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3046–3055, 2019.
- [103] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.
- [104] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit*. Kitware, 2006.
- [105] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015.
- [106] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *International Conference on Neural Information Processing Systems (NeurIPS)*, page 41–48, 2003.
- [107] B. S. Shetty and P. Shetty. *Digital Forensic Science*. IntechOpen, 2020.
- [108] D. Shi, M. Orouskhani, and Y. Orouskhani. A conditional triplet loss for few-shot learning and its application to image co-segmentation. *Neural Networks*, 137:54–62, 2021.

- [109] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.
- [110] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR*, 2015.
- [111] M. Singh, S. Nagpal, R. Singh, M. Vatsa, and A. Noore. Learning a shared transform model for skull to digital face image matching. In *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–7, 2018.
- [112] Stanford Vision and Learning Lab. Convolutional neural networks (cnns / convnets). <https://cs231n.github.io/convolutional-networks/#conv>, 2021.
- [113] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.
- [114] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2021.
- [115] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 1701–1708, 2014.
- [116] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 6105–6114, 2019.
- [117] H. Taniai. keras-facenet. <https://github.com/nyoki-mtl/keras-facenet>, 2018.
- [118] TensorFlow Developers. Tensorflow. <https://www.tensorflow.org/>, 2022.
- [119] T. Thompson and S. Black. *Forensic human identification: an introduction*. CRC Press, 2006.
- [120] D. H. Ubelaker. *Forensic Science: Current Issues, Future Directions*. John Wiley & Sons, Incorporated, 2012.

- [121] D. H. Ubelaker. *FORENSIC ANTHROPOLOGY*, chapter 2, pages 43–71. John Wiley & Sons, Ltd, 2018.
- [122] D. H. Ubelaker, Y. Wu, and Q. R. Cordero. Craniofacial photographic superimposition: New developments. *Forensic Science International (FSI)*, 1:271–274, 2019.
- [123] United Nations. Cerca de 1,35 millones de personas murieron en los últimos 20 años debido a desastres naturales. <https://news.un.org/es/story/2016/10/1366641>, 2016.
- [124] A. Valsecchi, S. Damas, and O. Cordón. A robust and efficient method for skull-face overlay in computerized craniofacial superimposition. *IEEE Transactions on Information Forensics and Security (TIFS)*, 13(8):1960–1974, 2018.
- [125] B. Wang, W. Gao, and B. Yang. Recognition method and application of wild vegetables based on lightweight convolutional neural network model. *Journal of Network Intelligence (JNI)*, 7(2):347–364, 2022.
- [126] D. Wang, C. Otto, and A. K. Jain. Face search at scale. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39:1122–1136, 2017.
- [127] F. Wang, J. Cheng, W. Liu, and H. Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters (SPL)*, 25:2926–930, 2018.
- [128] J. Wang, L. Yang, Z. Huo, W. He, and J. Luo. Multi-label classification of fundus images with efficientnet. *IEEE Access*, 8:212499–212508, 2020.
- [129] M. Wang and W. Deng. Deep face recognition: A survey. *Neurocomputing*, 429:215–244, 2021.
- [130] B. Yang, Q. Le, G. Bender, and J. Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.
- [131] M. Yani, S. Irawan, and C. Setianingsih. Application of transfer learning using convolutional neural network method for early detection of terry’s nail. *Journal of Physics: Conference Series (JPCS)*, 1201:012052, 2019.
- [132] G. Yao, T. Lei, and J. Zhong. A review of convolutional-neural-network-based action recognition. *Pattern Recognition Letters*, 118:14–22, 2019.

- [133] N. Yuvaraj, N. Kousik, R. A. Raja, and M. Saravanan. Automatic skull-face overlay and mandible articulation in data science by airs-genetic algorithm. *International Journal of Intelligent Networks (IJIN)*, 1:9–16, 2020.
- [134] Z. Zhang, Y. Song, and H. Qi. Age progression/regression by conditional adversarial autoencoder. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [135] J. Zhao, L. Xiong, P. Karlekar Jayashree, J. Li, F. Zhao, Z. Wang, P. Sugiri Pranata, P. Shengmei Shen, S. Yan, and J. Feng. Dual-agent gans for photorealistic and identity preserving profile face synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.
- [136] L. Zhu, X. Wen, L. Mo, J. Ma, and D. Wang. Robust location-secured high-definition image watermarking based on key-point detection and deep learning. *Optik*, 248:168194, 2021.

Apéndice A

Características de los modelos 3D craneales

Tabla 7.1: Información extendida relacionada con el conjunto de modelos 3D craneales utilizado. Los siguientes acrónimos se han definido para los rótulos de la tabla: S - Sexo, IF - Dispone de imagen facial, I3D - Información sobre el modelo 3D, N_V - número de vértices de su modelo 3D en miles de vértices, N_C el número de caras de su modelo 3D en miles de caras. Valores para la columna S: H - Hombre, M - Mujer, D - Desconocido. Para la columna IF: S - Si, N - No. Para la columna I3D: CC - Cráneo completo, CI- Cráneo completo e inicio de columna, SH - Sin hueso parietal y occipital, SM - Sin mandíbula, MS - Sin mandíbula y sobre soporte, ER - Mal modelo.

| PM | S | IF | I3D | N _V | N _C | PM | S | IF | I3D | N _V | N _C |
|----|---|----|-----|----------------|----------------|----|---|----|-----|----------------|----------------|
| 1 | D | N | CC | 187 | 366 | 60 | H | S | CC | 508 | 1016 |
| 2 | M | S | SH | 2682 | 4707 | 61 | M | S | CC | 214 | 428 |
| 3 | H | S | MS | 249 | 491 | 62 | H | S | MS | 298 | 591 |
| 4 | H | S | CC | 181 | 358 | 63 | D | N | CC | 416 | 833 |
| 5 | H | S | SH | 2530 | 5018 | 64 | D | N | CC | 177 | 349 |
| 6 | D | N | CC | 185 | 364 | 65 | M | S | MS | 250 | 494 |
| 7 | H | S | SH | 1111 | 2223 | 66 | D | N | CC | 310 | 621 |
| 8 | D | N | CC | 187 | 370 | 67 | D | S | CC | 200 | 394 |
| 9 | D | N | CC | 194 | 379 | 68 | D | N | CC | 185 | 365 |
| 10 | D | N | CC | 176 | 348 | 69 | M | S | CC | 328 | 645 |
| 11 | M | S | CC | 163 | 322 | 70 | D | N | CC | 174 | 342 |
| 12 | M | S | SH | 2824 | 4022 | 71 | H | S | SM | 361 | 692 |
| 13 | D | N | CC | 246 | 494 | 72 | H | S | CC | 464 | 927 |
| 14 | M | S | CC | 172 | 342 | 73 | M | S | MS | 282 | 559 |
| 15 | H | S | CC | 168 | 334 | 74 | H | S | CC | 265 | 530 |
| 16 | H | S | CC | 203 | 406 | 75 | M | S | CC | 282 | 565 |
| 17 | D | N | CC | 162 | 317 | 76 | M | S | CC | 215 | 432 |
| 18 | H | S | CC | 180 | 354 | 77 | M | S | MS | 262 | 520 |
| 19 | D | N | CC | 193 | 381 | 78 | H | S | CC | 404 | 799 |
| 20 | H | S | CC | 205 | 405 | 79 | D | N | CC | 180 | 355 |
| 21 | M | S | CC | 280 | 563 | 80 | H | S | CC | 389 | 778 |
| 22 | H | S | CC | 183 | 362 | 81 | M | S | SH | 2270 | 3902 |
| 23 | H | S | CC | 185 | 362 | 82 | H | S | CC | 207 | 414 |
| 24 | D | N | ER | — | — | 83 | M | S | CC | 243 | 480 |
| 25 | D | N | CC | 164 | 325 | 84 | M | S | SM | 359 | 713 |

Continúa en la siguiente página.

Tabla 7.1 – *Continuación de la anterior página.*

| PM | S | IF | I3D | N_V | N_C | PM | S | IF | I3D | N_V | N_C |
|-----------|----------|-----------|------------|----------------------|----------------------|-----------|----------|-----------|------------|----------------------|----------------------|
| 26 | D | N | CC | 167 | 329 | 85 | M | S | MS | 273 | 544 |
| 27 | H | S | CC | 193 | 380 | 86 | D | N | CC | 177 | 349 |
| 28 | D | N | CC | 184 | 361 | 87 | D | N | SM | 152 | 298 |
| 29 | H | S | MS | 282 | 558 | 88 | H | S | MS | 226 | 448 |
| 30 | D | N | CC | 180 | 355 | 89 | H | S | SM | 419 | 828 |
| 31 | H | S | SH | 2547 | 4499 | 90 | M | S | MS | 266 | 527 |
| 32 | D | N | CC | 186 | 369 | 91 | D | N | CC | 247 | 495 |
| 33 | M | S | SH | 1698 | 3313 | 92 | H | S | CC | 303 | 598 |
| 34 | H | S | CC | 192 | 380 | 93 | H | S | MS | 290 | 573 |
| 35 | D | N | CC | 171 | 338 | 94 | D | N | CI | 846 | 1700 |
| 36 | M | S | MS | 230 | 455 | 95 | D | N | CI | 960 | 1938 |
| 37 | M | S | CC | 62 | 119 | 96 | D | N | CI | 288 | 611 |
| 38 | D | N | CC | 184 | 363 | 97 | D | N | CI | 273 | 573 |
| 39 | H | S | CC | 425 | 849 | 98 | D | N | CI | 914 | 1846 |
| 40 | D | N | ER | — | — | 99 | D | N | CI | 852 | 1728 |
| 41 | D | N | CC | 168 | 331 | 100 | D | N | CI | 957 | 1926 |
| 42 | H | S | SH | 2410 | 4150 | 101 | D | N | CI | 831 | 1662 |
| 43 | M | S | CC | 291 | 582 | 102 | D | N | CI | 271 | 561 |
| 44 | D | N | CC | 193 | 378 | 103 | D | N | CI | 905 | 1823 |
| 45 | D | N | CC | 402 | 805 | 104 | D | N | CI | 188 | 391 |
| 46 | H | S | CC | 173 | 342 | 105 | D | N | CI | 906 | 1824 |
| 47 | M | S | SH | 2146 | 4253 | 106 | D | N | ER | — | — |
| 48 | D | N | CC | 121 | 243 | 107 | D | N | CI | 818 | 1643 |
| 49 | D | N | CC | 196 | 386 | 108 | D | N | CI | 276 | 590 |
| 50 | D | N | CC | 176 | 347 | 109 | D | N | CI | 185 | 401 |
| 51 | H | S | CC | 431 | 853 | 110 | D | N | CI | 778 | 1572 |
| 52 | D | N | CC | 248 | 496 | 111 | D | N | CI | 773 | 1560 |
| 53 | H | S | MS | 290 | 575 | 112 | D | N | CI | 899 | 1814 |
| 54 | D | N | CC | 361 | 722 | 113 | D | N | CI | 866 | 1743 |
| 55 | M | S | CC | 258 | 517 | 114 | D | N | CI | 865 | 1742 |
| 56 | M | S | CC | 308 | 306 | 115 | D | N | CI | 1053 | 2143 |
| 57 | H | S | CC | 337 | 667 | 116 | D | N | CI | 277 | 582 |
| 58 | H | S | CC | 678 | 1357 | 117 | D | N | CI | 722 | 1421 |
| 59 | D | N | CC | 191 | 377 | | | | | | |

Apéndice B

Algoritmos detallados para la generación online de tripletas

Estrategia de generación online de tripletas batch-all

El procedimiento seguido es el siguiente, que incluye la generación de las tripletas válidas (eliminando las tripletas sencillas) y el cómputo de la función de pérdida sobre todas ellas:

1. Recibir un batch con tres secuencias, I_A , I_P y I_N , de imágenes ancla, positivo y negativo respectivamente, con tamaño $tambatch$.
2. Obtener los *embeddings*, E_A , E_P y E_N , de las imágenes de las tres secuencias I_A , I_P y I_N .
3. Computar la distancia entre cada *embedding* ancla y cada *embedding* positivo y la distancia entre cada *embedding* ancla y cada *embedding* positivo, obteniendo dos matrices de distancias $D_{A,P}$ y $D_{A,N}$, respectivamente. Ambas matrices tienen dimensión $tambatch \times tambatch$, donde el elemento $D[i, j]$ se corresponde con la distancia entre el *embedding* facial i y el *embedding* craneal j (positivo o negativo, dependiendo de la matriz) de las secuencias de entrada del batch.
4. Computar la función de pérdida Triplet Loss para todas las posibles tripletas del batch: cada imagen facial se combina con todas las imágenes craneales positivas y cada uno de estos pares se combina, a su vez, con todas las imágenes craneales negativas. Se obtiene una matriz F de tamaño $tambatch \times tambatch \times tambatch$, donde $F[i, j, k]$ se corresponde con el valor de la función de pérdida de la tripleta compuesta por la imagen facial i , la imagen positiva j y la imagen negativa k de las secuencias de entrada del batch.
5. Eliminar las tripletas incorrectas, es decir, aquellas donde la imagen facial no se corresponde con el ejemplo positivo (su etiqueta es diferente y, por tanto, pertenecen a individuos diferentes) y aquellas donde la imagen facial o el ejemplo positivo pertenecen al mismo individuo que el ejemplo negativo. Esto se traduce en poner a 0 los elementos $F[i, j, k]$ donde $l_A[i] \neq l_P[j] \vee l_A[i] = l_N[k]$, siendo l_A, l_P, l_N las secuencias de etiquetas de las imágenes.
6. Computar la función de pérdida del batch, constituida por la media de las funciones de pérdida de las tripletas válidas. Para ello basta con

obtener la suma de los valores de F y dividir por el número de tripletas válidas. Tanto las tripletas sencillas como las incorrectas tienen pérdida 0, por lo que ni influyen en el anterior cálculo.

Estrategia de generación online de tripletas batch-hard

El procedimiento seguido para crear las tripletas más complicadas y obtener la función de pérdida del batch sobre ellas es el siguiente:

1. Recibir un batch con tres secuencias, I_A , I_P y I_N , de imágenes ancla, positivo y negativo respectivamente, con tamaño tam_{batch} .
2. Obtener los *embeddings*, E_A , E_P y E_N , de las imágenes de las tres secuencias I_A , I_P y I_N .
3. Computar la distancia entre cada *embedding* ancla y cada *embedding* positivo y la distancia entre cada *embedding* ancla y cada *embedding* positivo, obteniendo dos matrices de distancias $D_{A,P}$ y $D_{A,N}$, respectivamente. Ambas matrices tienen dimensión $tam_{batch} \times tam_{batch}$, donde el elemento $D[i, j]$ se corresponde con la distancia entre el *embedding* facial i y el *embedding* craneal j (positivo o negativo, dependiendo de la matriz) de las secuencias de entrada del batch.
4. Eliminar los pares de *embeddings* incorrectos: aquellos ancla-positivo cuyas etiquetas sean diferentes ($l_A[i] \neq l_P[j]$) y aquellos ancla-negativo cuyas etiquetas sean iguales ($l_A[h] = l_N[k]$), siendo l_A, l_P, l_N las secuencias de etiquetas de las imágenes. Esto se traduce en poner a 0 los elementos $D_{A,P}[i, j]$ (ya que se desea obtener el máximo de las distancias positivas, para encontrar el ejemplo positivo más complicado) y en poner a *infinito* los elementos $D_{A,N}[h, k]$ (ya que se desea obtener la mínima de las distancias negativas, para encontrar el ejemplo negativo más complicado).
5. Obtener el ejemplo positivo más complicado como el máximo de $D_{A,P}$ y el ejemplo negativo más complicado como el mínimo de $D_{A,N}$ y calcular la función de pérdida Triplet Loss sobre ellos (aplicando la Ecuación 2.9).

Apéndice C

Algoritmos detallados para el test de los modelos

Algoritmo de cómputo de los rankings en Test Positivo

El procedimiento de cómputo de los rankings en Test Positivo es el siguiente, en el que se reciben las imágenes craneales positivas y se devuelve el ranking de cada una de ellas:

1. Recibir las imágenes craneales positivas de test, \mathbf{S} (con tam_s imágenes), las imágenes faciales de la BD de test, \mathbf{F} (con tam_f imágenes) y sus respectivas etiquetas (individuos), \mathbf{l}_S y \mathbf{l}_F .
2. Computar la distancia entre cada imagen craneal de S y cada imagen facial de F , obteniendo una matriz \mathbf{D} (dimensionalidad $tam_s \times tam_f$).
 - El elemento $D[i, j]$ ($i \in \{1, tam_s\}, j \in \{1, tam_f\}$) se corresponde con la distancia entre la imagen $S[i]$ y la imagen $F[j]$.
 - En el caso de la SNN con TL se deben obtener los *embeddings* de S y los de F y luego calcular la distancia entre todos ellos. A menor distancia, mayor es la semejanza entre las dos imágenes.
 - En el caso de la SNN con salida sigmoidal se debe obtener la salida (distancia) de la red para todos los posibles pares de imágenes entre S y F . A mayor distancia, mayor semejanza entre imágenes.
3. Calcular los índices de ordenación de los elementos por filas de la matriz D , obteniendo la matriz \mathbf{D}_{ord} (dimensionalidad $tam_s \times tam_f$).
 - La fila $D_{ord}[i]$ ($i \in \{1, tam_s\}$) contiene la secuencia de índices que ordenan de mayor a menor semejanza los elementos fila i de la matriz D (en la SNN con TL, de menor a mayor distancia; en la SNN con salida sigmoidal, de mayor a menor distancia),
 - $D_{ord}[i, 0]$ es el índice del elemento con mayor semejanza en la fila $D[i]$.
 - La predicción de la imagen craneal $S[i]$ es la imagen facial $D_{ord}[i, 0]$ (siempre que la distancia $D[i, D_{ord}[i, 0]]$ sea mayor que el umbral de decisión).
4. Utilizar D_{ord} como una máscara para indexar l_f en cada fila, obteniendo la matriz de predicciones \mathbf{P} (dimensionalidad $tam_s \times tam_f$).

- La fila i ($i \in \{1, tam_s\}$) de la matriz P contiene las etiquetas predichas ordenadas de mayor a menor semejanza para la imagen craneal $S[i]$.
 - El elemento $P[i, j]$ ($i \in \{1, tam_s\}, j \in \{1, tam_f\}$) se corresponde con la etiqueta de la imagen facial predicha en el ranking j para la imagen craneal $S[i]$.
 - Cada elemento $P[i, j]$ es único en la fila i de la matriz P .
5. Obtener una máscara binaria, \mathbf{M} (dimensionalidad $tam_s \times tam_f$), tal que $M[i, j] = 1 \leftrightarrow P[i, j] = l_s[i]$ ($i \in \{1, tam_s\}, j \in \{1, tam_f\}$).
 - Solo existe un 1 en cada fila i de la matriz M .
 6. Obtener los rankings, \mathbf{R} (tamaño tam_s), de las imágenes craneales (sin tener en cuenta el umbral de decisión) como el índice de los máximos por filas de la matriz M .
 - La distancia $D[i, R[i]]$ es la distancia entre la imagen craneal $S[i]$ y su imagen facial correspondiente (la perteneciente al mismo individuo).
 7. Incrementar en una posición aquellos rankings donde la distancia entre la imagen craneal y su imagen facial correspondiente no sea lo suficientemente semejante, en función al umbral de decisión.
 - Para la SNN con TL, se incrementan aquellos donde la distancia es **mayor** que el umbral de decisión (mayor distancia implica menor semejanza).
 - Para la SNN con salida sigmoidal, se incrementan aquellos donde la distancia es **menor** que el umbral de decisión (menor distancia implica menor semejanza).
 8. Devolver los rankings \mathbf{R} de las imágenes craneales positivas de test.

Algoritmo de cómputo de los NFA en Test Negativo

El procedimiento de cómputo de los valores NFA en Test Negativo es el siguiente, en el que se reciben las imágenes craneales negativas y se devuelve el NFA de cada una de ellas:

1. Recibir las imágenes craneales negativas de test, \mathbf{S} (con tam_s imágenes), las imágenes faciales de la BD de test, \mathbf{F} (con tam_f imágenes) y sus respectivas etiquetas (individuos), \mathbf{l}_S y \mathbf{l}_F .
2. Computar la distancia entre cada imagen craneal de S y cada imagen facial de F , obteniendo una matriz \mathbf{D} (dimensionalidad $tam_s \times tam_f$).

- El elemento $D[i, j]$ ($i \in \{1, tam_s\}, j \in \{1, tam_f\}$) se corresponde con la distancia entre la imagen $S[i]$ y la imagen $F[j]$.
 - En el caso de la SNN con TL se deben obtener los *embeddings* de S y los de F y luego calcular la distancia entre todos ellos. A menor distancia, mayor es la semejanza entre las dos imágenes.
 - En el caso de la SNN con salida sigmoidal se debe obtener la salida (distancia) de la red para todos los posibles pares de imágenes entre S y F . A mayor distancia, mayor semejanza entre imágenes.
3. Obtener una máscara binaria, M (dimensionalidad $tam_s \times tam_f$), tal que $M[i, j] = 1$ si $D[i, j]$ es suficientemente semejante como para ser emparejada, en función del umbral de decisión β ($i \in \{1, tam_s\}, j \in \{1, tam_f\}$).
 - En el caso de la SNN con Tl, la distancia es suficientemente semejante si $D[i, j] < \beta$.
 - En el caso de la SNN con salida sigmoidal, la distancia es suficientemente semejante si $D[i, j] > \beta$.
 4. Obtener los **NFA** como la suma por filas de los elementos de la matriz M .
 - $NFA[i]$ es el número de 1 en la fila i de la matriz M .
 5. Devolver los **NFA** de las imágenes craneales negativas de Test.