uc3m | Universidad Carlos III de Madrid

Doble Grado en Administración y Dirección
de Empresas e Ingeniería Informática
2019-2020

Trabajo Fin de Grado

"Redes Neuronales Convolucionales Siamesas aplicadas a la Verificación Facial"

Asier Alcaide Martínez

Tutores

Antonio Berlanga de Jesús
Miguel Ángel Patricio Guisado

Colmenarejo, 2020

RESUMEN

En este trabajo de fin de grado se realiza un análisis sobre el diseño e implementación de las técnicas más potentes en biometría facial aplicadas a dispositivos móviles. Se propone y explica una nueva técnica utilizada para el diseño de las redes de neuronas convolucionales siamesas, muy eficiente para este tipo de problemas, utilizando el estado del arte de distintas arquitecturas que se analizan, y la implementación de cada una de ellas, obteniendo resultados que permitirán cumplir con los objetivos propuestos tanto de tiempos de ejecución como de exactitud y precisión.

Durante el desarrollo, se han propuesto cuatro de las arquitecturas utilizadas durante todo el proceso de implementación, explicando cada una de ellas y analizando sus pros y contras para este problema.

Más adelante se analizan los resultados de cada una de ellas, poniendo en contexto el resto de tecnologías utilizadas hasta el momento y realizando conclusiones sobre ellas.

Palabras clave

Inteligencia artificial; biometría facial; verificación biométrica; redes de neuronas artificiales; redes de neuronas convolucionales; redes de neuronas convolucionales siamesas; clasificación binaria.

DEDICATORIA

Me gustaría expresar mi agradecimiento a todas aquellas personas que han logrado hacer de este proyecto un gran cambio en muchos aspectos de mi vida tanto personal como profesional.

En primer lugar, agradecer a toda la cátedra BQ-UC3M y al fantástico Grupo de Inteligencia Artificial Aplicada (GIAA) de Colmenarejo, donde no he podido estar más contento con profesores como Antonio y Miguel Ángel, que me han dado las llaves de todas las puertas sobre el mundo del aprendizaje automático, además del equipo de alumnos formados por Daniel González, Juan Abascal y Luis Víctor Hevia. Gracias por todas los pequeños avances y conocimientos que hemos podido compartir juntos. Por supuesto, agradecer también al equipo de "bq" formado por grandes mentes como Mikel Fernández, Ravin Dhalani, o Carlos Iniesta, sin ellos todo esto no se hubiera llevado a cabo.

Quiero agradecer, por otro lado, a todas las personas que me han apoyado a lo largo de estos dos últimos años, haciendo de este trabajo realidad.

La familia, por un lado, aquella que ha permitido eliminar cualquier limitación, apoyando tanto financiera como emotivamente mis estudios hasta el último de sus esfuerzos. Los amigos, por otro lado, aquellos que a pesar de las distancias siempre han estado cerca, escuchando y apoyando en todo momento. Mi chica, por último, aquella que ha soportado desde incontables reencuentros a través de una pantalla, hasta meses de intensa cuarentena que todos conocemos.

Gracias a todos.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

A lo largo de todo este documento se ha propuesto solucionar un problema de biometría facial mediante técnicas de Aprendizaje Profundo o "Deep Learning". Existen numerosas definiciones de este concepto de la biometría. Según Woodward et al. [1], la biometría se puede definir como "el reconocimiento automático de personas utilizando distintos medios.". Sin embargo, otros autores consideran la biometría como el reconocimiento de rasgos físicos de cualquier ser vivo.

Considerando la biometría como aquella que analiza los comportamientos físicos o intrínsecos de los seres humanos, así como las características de comportamiento, existen diversas técnicas que permiten extraer información de personas. Cualquier rasgo característico puede ser utilizado para poder reconocer y distinguirnos entre nosotros: reconocimiento facial mediante patrones o imágenes, mediante la retina o iris de los ojos, geometrías o formas de la mano o venas (geometría dactilar y vascular), reconocimiento de voz, de firma, basado en teclear, etc.

El uso de la biometría ha aumentado considerablemente en los últimos años. Su origen hasta donde se conoce viene en el siglo XIV en China, donde los comerciantes utilizaban papel y tinta para imprimir las palmas de las manos. En Occidente, no fue hasta el siglo XIX cuando se comenzó a utilizar medidas específicas del cuerpo y cabeza para identificar criminales [2].

Existen numerosas aplicaciones biométricas actualmente que ayudan y mejoran la calidad de vida de las personas, dependiendo de la técnica que se utiliza. La identificación de ciudadanos permite el control de todas las personas en un estado de forma más fácil; el control de accesos a recintos mediante huella dactilar o reconocimiento facial agiliza la entrada y salida de los mismos; firmas digitales o pagos bancarios mediante biometría facilitan también la rápida accesibilidad a herramientas de software; y por supuesto la investigación de delitos permite mejorar su precisión y certeza gracias a la biometría. Además, empresas que utilizan esta tecnología adquieren una gran ventaja competitiva frente al resto de compañías, permitiendo aportar un valor extra al producto o servicio en cuestión.

Todas estas aplicaciones son algunas de las más destacadas en el entorno de la biometría. Unas de las técnicas más utilizadas a lo largo de la última década ha sido el reconocimiento biométrico facial.

1.1. BIOMETRÍA FACIAL

La biometría facial es un mecanismo biométrico específico que permite determinar la identidad de una persona analizando su rostro [1]. El rostro es una característica física que permite distinguir e identificar personas con gran precisión, y gracias a la biometría facial es posible realizar este tipo de identificación sin necesidad de realizar ningún contacto físico o la colaboración por parte del usuario, como ocurre en la biometría dactilar. Además, otro aspecto positivo de la biometría facial frente a la dactilar es la seguridad: considerando modelos robustos y precisos, una huella dactilar puede ser robada cuando la persona afectada está dormida o desmayada, mientras que el reconocimiento facial suele exigir el hecho de tener los ojos abiertos y mantener una expresión natural del rostro.

Distinguimos dos principales tipos de reconocimiento facial:

- Reconocimiento facial en dos dimensiones: utilizado mucho antes que el reconocimiento en tres dimensiones, se analizan una o varias imágenes (vídeo) 2D para obtener resultados.
- Reconocimiento en tres dimensiones: utilizan tecnologías como la infrarroja para elaborar un rostro virtual tridimensional más parecido al de una persona.

Existen además variaciones entre estos dos tipos, como puede ser la construcción tridimensional de un rostro mediante vídeo o secuencias de imágenes [3]; o el reconocimiento de patrones en la textura de la piel [4], que han permitido desarrollarse positivamente gracias al aumento computacional de estos últimos años.

1.2. MOTIVACIÓN DEL TRABAJO

La inteligencia artificial es uno de los conceptos más hablados por las empresas hoy en día, concretamente el reconocimiento de patrones y "Machine Learning" o Aprendizaje Automático. Son tecnologías que han podido explotarse en gran medida a lo largo de estos últimos años, y que han permitido grandes avances en diversos ámbitos.

La biometría facial a través de técnicas como "Machine Learning" permite aplicar distintas herramientas muy potentes a la hora de realizar verificaciones faciales de personas. La gran motivación del estudio elaborado consiste en comprender de una manera profunda el comportamiento del "Deep Learning" en reconocimiento y verificación facial, por medio de las herramientas más utilizadas del momento, combinando una mejora en los resultados de estas verificaciones además del estudio del rendimiento de las mismas en dispositivos con limitada capacidad computacional.

1.3. MARCO REGULADOR

Para realizar modelos de Machine Learning supervisados, es necesario utilizar una batería de datos ("dataset"), en este caso imágenes, con sus respectivas etiquetas o "labels".

Un "dataset" de reconocimiento biométrico facial está basado en su práctica totalidad por imágenes reales de personas. Esto puede resultar un gran problema de aspecto legal. Tratar de utilizar sin consentimiento los derechos de imagen de las personas desencadena en un grave problema de derechos de imagen y de privacidad. A continuación, se describe el marco regulador que afecta al uso de imágenes privadas sin consentimiento:

- El artículo 18 de la Constitución Española establece que "Se garantiza el derecho al honor, a la intimidad personal y familiar y a la propia imagen. […]"
- El artículo primero de la Ley Orgánica 1/1982 establece que: "El derecho al honor, a la intimidad personal y familiar y a la propia imagen es irrenunciable, inalienable e imprescriptible. […]"
- En el artículo séptimo de la Ley Orgánica 1/1982 establece que: "Tendrán la consideración de intromisiones ilegítimas en el ámbito de protección delimitado por el artículo segundo de esta Ley: […] Seis. La utilización del nombre, de la voz o de la imagen de una persona para fines publicitarios, comerciales o de naturaleza análoga."

- El nuevo Reglamento Europeo de Protección de Datos (RGPD) considera la imagen y la voz como un dato de carácter personal y sujeto, por tanto, a protección. Para poder tratar cualquier dato personal se exige un consentimiento expreso a cada una de las personas:
    - Establece que "Quedan prohibidos el tratamiento de datos personales que revelen el origen étnico o racial, las opiniones políticas, las convicciones religiosas o filosóficas, o la afiliación sindical, y el tratamiento de datos genéticos, **datos biométricos** dirigidos a identificar de manera unívoca a una persona física, datos relativos a la salud o datos relativos a la vida sexual o la orientación sexual de una persona física.". No obstante, existe la excepción en caso donde el usuario acepta el tratamiento de los datos como se muestra más adelante.
    - Establece que no está permitido el tratamiento de estas imágenes *"*cuando el hecho de ser tratadas con medios técnicos específicos permita la identificación o la autenticación unívocas de una persona física.". Todo ello puede ser excepción en función de las leyes que cada Estado miembro haya podido crear en situaciones específicas.
    - La RGPD además indica que el tratamiento de datos personales biométricos es permitido siempre y cuando sus fines sean relacionados con la salud en beneficio de las personas físicas y de la sociedad en su conjunto, por parte de las autoridades gestoras de la salud.
    - No obstante, todas las citas anteriores pueden ser permitidas siempre y cuando ocurran algunas situaciones, entre ellas cuando "**el interesado dio su consentimiento explícito para el tratamiento de dichos datos personales con uno o más de los fines especificados**, excepto cuando el Derecho de la Unión o de los Estados miembros establezca que la prohibición mencionada en el apartado 1 no puede ser levantada por el interesado;".
        - Los datos protegidos "se podrán utilizar si fuese necesario para proteger **intereses vitales** del interesado o de otra persona física, si el interesado no estuviera capacitado para dar su consentimiento."
        - También se podrán utilizar "cuando el tratamiento se refiriera a datos personales que el interesado hubiese hecho anteriormente **manifiestamente públicos."**
        - También para fines de medicina preventiva o laboral.
    - El RGPD se debe considerar en su práctica totalidad para poder realizar las distintas operaciones y usos sobre los datos de personas. Para la utilización de fotografías por parte de una compañía, si el usuario es mayor de 14 años, será totalmente imprescindible contar con el consentimiento expreso del mismo. En caso de que la persona sea menor de 14 años, deberá aceptar expresamente el consentimiento los padres o tutores legales correspondientes. Todo ello deberá ser impuesto por un documento elaborado por parte de la empresa interesada, que deberá ser aceptado expresamente. En caso de no cumplir con la normativa, las infracciones podrán alcanzar desde los 40001 hasta los 300000 euros.

Por tanto, a nivel europeo, bajo las diversas leyes creadas sobre la protección de datos, el tratamiento de imágenes para la verificación facial **se deberá dar estrictamente el consentimiento explícito de las personas que hayan sido sometidas a dicho proceso**. Para la elaboración de este trabajo, las imágenes han sido elaboradas a partir de un conjunto de datos formado por personajes públicos, donde en la totalidad son imágenes obtenidas a partir de fotografías vistas por cualquiera que quiera verlas.

En base a todo lo descrito previamente, se concluye el marco regulador como:

1. El uso de imágenes en la fase de entrenamiento de modelos, las imágenes procesadas han sido las obtenidas por el conjunto de datos LFW, de carácter público debido a los personajes públicos formados en él.
2. El uso de imágenes para biometría facial, según la LGPD, constituye entre el grupo de "datos sensitivos", y por tanto no se permite como norma general el tratamiento de ellas. No obstante**, sí que pueden ser tratadas en el caso donde exista consentimiento explícito de las personas involucradas**, si existen obligaciones legales que permitan su uso, o si se requiere el procesamiento como interés público. Por tanto, en el caso de la fase de predicción o de verificación de individuos, cada usuario de la aplicación móvil deberá explícitamente dar su consentimiento.
3. En cuanto al tratamiento de datos sensibles destinados a la "nube", la implementación de estos modelos ha permitido realizar las predicciones en los propios dispositivos de manera local, sin tener que ser enviados a servidores ajenos, ya que el modelo será previamente entrenado y se instalará posteriormente en los dispositivos, realizando comparaciones de imágenes en cada uno de ellos de manera independiente.

1.4. IMPACTO SOCIO-ECONÓMICO

La biometría facial es un campo en auge con un crecimiento positivo año tras año. Cada vez más empresas tratan de instalar sus productos tanto en teléfonos inteligentes como en otros tipos de dispositivos, como pueden ser cajeros automáticos, cámaras de seguridad de recitos probados, o incluso de recintos públicos que requieran de alto nivel de seguridad y control de tráfico de personas (aeropuertos, etc.). La tecnología propuesta en este proyecto permite crear una ventaja competitiva frente a muchos de sus competidores, manteniendo o mejorando la tasa de falsos intrusos o falsos positivos y disminuyendo los tiempos ejecución de cada verificación. Esta ventaja competitiva puede generar un gran impacto en el valor esperado del producto o servicio ofrecido, facilitando la capacidad de venta gracias a una calidad-precio muy favorable.

Por otro lado, este tipo de tecnologías pueden generar un gran impacto tanto positivo como negativo en la sociedad.

- Un buen uso de las mismas puede facilitar la vida de muchas personas, mejorando la seguridad a través de múltiples ejemplos como la protección de desbloqueo de teléfonos móviles, detección de personas criminales en lugares de riesgo, o en una situación de pandemia como lo que se ha producido en 2020, a través de desbloqueo y

acceso a espacios privados del usuario sin necesidad de contacto físico, como es el caso de cajeros automáticos.

- Sin embargo, un uso incorrecto de las mismas a nivel ético puede generar graves violaciones de privacidad de personas en muchas situaciones, como puede ser el control y seguimiento del usuario a través de cámaras, como es el caso de China. Además, podría existir una mala aplicación para fines militares o actos de violencia, a través de detección y confirmación de personas.

En definitiva, la aplicación de la tecnología estudiada, desarrollada e implementada en este trabajo permite llegar a cabo una aplicación práctica potencialmente viable, aunque es necesario no olvidar los aspectos éticos y sociales que pueden influir en nuestra sociedad. Para ver el análisis de planificación y presupuesto desarrollado en este trabajo, se deberá avanzar hasta el capítulo de *Planificación y Presupuesto*.

2. OBJETIVOS

El objetivo principal de este trabajo es analizar, diseñar e implementar un nuevo modelo de "machine Learning" como parte de un sistema de verificación biométrica facial para dispositivos móviles.

Este objetivo principal puede descomponerse en los siguientes objetivos parciales:

- Analizar las técnicas actuales implementadas para modelos con este tipo de problemas de biometría facial.
- Diseñar e implementar una arquitectura de un modelo capaz de mejorar las técnicas actuales:
    - Aumentando la seguridad ante intrusos, basándose en una reducción de casos de falsos positivos.
    - Reduciendo los tiempos de ejecución de cada verificación, aprovechando limitaciones que existen en dispositivos móviles frente a otros sistemas.

El estudio tendrá que elaborarse a partir de un análisis previo de las tecnologías que se están aplicando en la actualidad, para posteriormente proponer un modelo que permita mejorar los posibles resultados que se obtienen.

En cuando al diseño e implementación del modelo, el estudio estará basado en un tipo de arquitectura de redes de neuronas llamada "Redes Convolucionales Siamesas". Se utilizará este tipo de arquitectura, y se realizarán distintas comparaciones con distintos modelos con el fin de obtener unos resultados globales beneficiosos en comparación con una arquitectura tradicional de técnicas de verificación facial. En base a ello, se intentarán lograr estos objetivos propuestos, permitiendo en su finalidad el desarrollo de una técnica innovadora y con grandes resultados en el campo de la biometría facial.

3. ESTADO DEL ARTE

3.1. PROBLEMA

Ya se ha mencionado anteriormente que la verificación facial no es perfecta, y a lo largo de estos últimos años se han dado múltiples casos de robos de privacidad gracias a pequeñas imperfecciones en estas "nuevas" tecnologías. Concretamente, se han dado casos donde existen grandes inexactitudes en la precisión de los falsos positivos, esto es, la cantidad de personas que se verifican como las mismas cuando en realidad no lo son, como es en el caso del pasado año donde el Gobierno Estadounidense realizó un estudio donde esta tasa era cien veces más alta en personas de origen afroamericano y asiáticos que en rostros de origen caucásico [5].

Además, la exactitud de estos modelos de reconocimiento y verificación facial están limitados en dispositivos con reducida capacidad computacional, donde los tiempos de ejecución pueden ser más lentos si se desean unos resultados altamente seguros.

Por ello, un aumento de la precisión en los modelos de verificación provocará una mejora directa en la seguridad y privacidad de estas aplicaciones. Con este estudio se propone solucionar este problema que existe actualmente, tratando de mejorar la precisión y exactitud de sistemas de biometría facial reduciendo los tiempos de ejecución en dispositivos ligeros y limitados, como pueden ser teléfonos móviles, cajeros automáticos, videocámaras con pequeños procesadores, etc.

3.2. SITUACIÓN ACTUAL

Cabe destacar que la *verificación facial* no tiene el mismo significado que el *reconocimiento facial.* El *reconocimiento facial* se basa en comparar una sola imagen de una persona con múltiples de ellas que están guardadas e identificadas previamente en algún almacenamiento de datos, con el fin de "averiguar" qué persona es la de la imagen en función de la similitud entre las guardadas anteriormente. Por otro lado, la *verificación facial* compara una sola imagen con otra, con el fin de "averiguar" si ambas imágenes consisten en la misma persona o no.

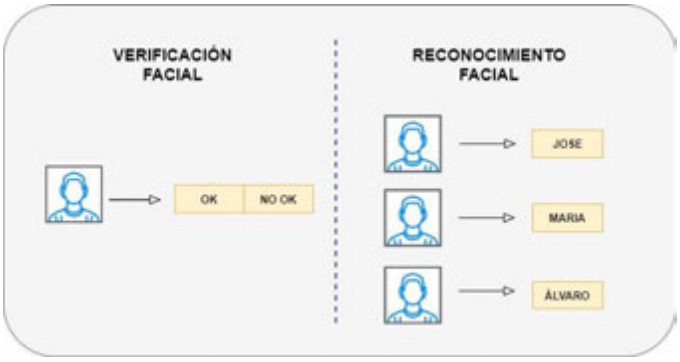


Ilustración 1: Verificación y reconocimiento facial. Fuente: elaboración propia.

La verificación facial, junto con el incremento computacional de los últimos años, ha permitido grandes avances para la mejorar el incremento continuo de métodos estadísticos más poderosos. Para lograr tales avances, esta técnica ha sufrido varios cambios hasta llegar al punto en el que nos encontramos hoy en día.

3.2.1. ORÍGENES

A lo largo de la historia de la biometría, el reconocimiento facial surgió antes de la verificación facial. Fue en la década de los 60 cuando un matemático americano diseñó un sistema para reconocer imágenes de personas manualmente [6]. Posteriormente se desarrollaron nuevas técnicas que mejoraron la precisión del anterior modelo, pero no fue hasta 1991 cuando se implementó por primera vez un modelo en tiempo real de reconocimiento facial, elaborado por Matthew Turk y Alex Pentland [7], donde utilizaron auto vectores que describían los rasgos físicos faciales.

En cuando a la verificación facial, en un primer momento se utilizaban técnicas tradicionales tales como la aplicación de filtros de correlación para comparar rostros [8]; más adelante fueron aplicadas distintas combinaciones como el reconocimiento facial y reconocimiento ocular a través del iris [9]; también se elaboraron otras arquitecturas como redes bayesianas basadas en la edad de los usuarios como característica adicional para verificación [10] o combinaciones de técnicas para la extracción de características faciales a través de algoritmos de optimización como AdaBoost y ondas de Gabor gracias a las transformadas de Fourier [11]. En 2009 Kumar et. Al. [12] publicaron un modelo con resultados muy destacados, que consistía en la aplicación de múltiples clasificadores binarios para cada uno de los atributos y características del rostro humano, obteniendo un resultado final a partir de la total combinación todos los clasificadores binarios.

3.2.2. REDES DE NEURONAS

Las redes de neuronas consisten en la unión de varias neuronas artificiales o perceptrones, que permiten realizar inferencias a través del comportamiento de los mismos. Los perceptrones simples [13] son la unidad básica de las redes de neuronas artificiales, donde reciben varios inputs de entrada y devuelven uno o varios de salida. El concepto se basa en un discriminador lineal, a partir del cual se desarrolla un algoritmo capaz de separar por subgrupos los distintos elementos en función de los inputs establecidos.

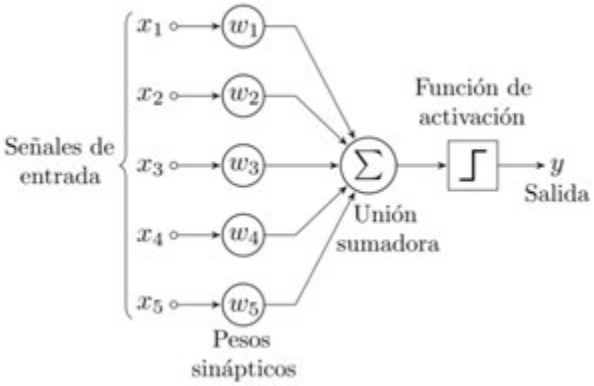


Ilustración 2: Diagrama de un perceptrón con cinco señales de entrada. [14]

En un primer momento, el perceptrón simple servía para clasificar problemas linealmente separables, clasificación que ya se podía hacer mediante métodos estadísticos, y de una forma mucho más eficiente. En 1986 surgieron las primeras redes de neuronas artificiales, formadas por la superposición de varias capas de perceptrones. A su vez surgió además el concepto de retro propagación o "*backpropagation"*, un método que permitía corregir parámetros en los perceptrones desde las últimas capas hasta las más cercanas a los inputs en función de los resultados esperados ya conocidos previamente.

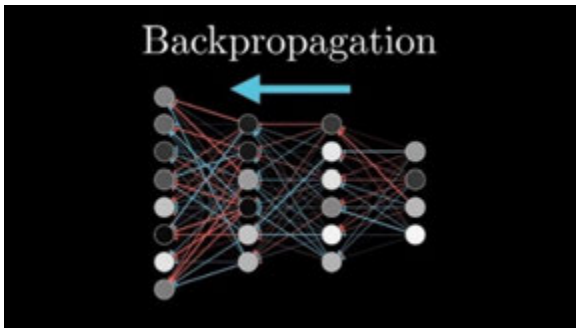


Ilustración 3: Ejemplo de red de neuronas artificial con retro-propagación [15].

Más tarde, a partir de la segunda década de este siglo, con el fin de las grandes limitaciones computacionales que existían hasta el momento, comenzaron a aplicarse nuevos modelos basados en el concepto de Deep Learning, entre ellos el comienzo de la explotación de las Redes de Neuronas Convolucionales.

3.2.3. REDES DE NEURONAS CONVOLUCIONALES

El concepto de convolucional marca la diferencia entre las redes de neuronas totalmente conectadas entre sí (o *fully connected* a partir de ahora) en que la salida de los perceptrones entre cada capa depende solamente de un grupo de "neuronas" de la anterior capa, y no de todas las anteriores a ella. Esto permite una reducción del coste computacional, al verse

reducido la cantidad de “pesos” utilizados, y la mejora en la regularización del modelo, evitando un posible *overfitting*.

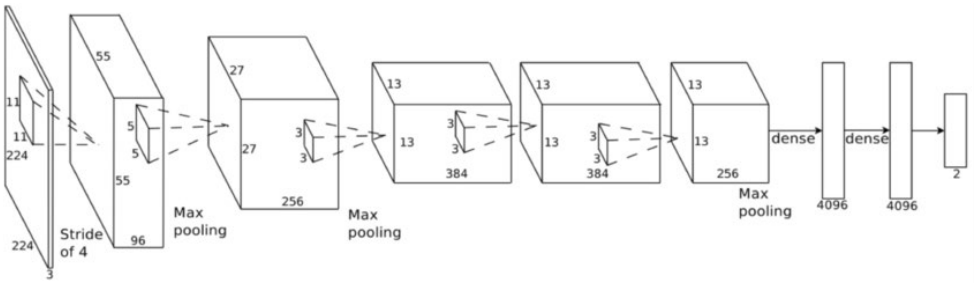


Ilustración 4. Ejemplo de AlexNet, una Red Neruronal Convolucional [16].

Las redes de neuronas tradicionales no escalan bien con imágenes. CIFAR-10 es un pequeño dataset compuesto por 60000 imágenes donde cada una de ellas forma 32x32x3 entradas a la red (32 ancho, 32 alto, 3 canales de colores), por lo que una única neurona entera conectada en una primera capa tendría 32x32x3=3072 pesos. Esta cantidad, a pesar de ser alta, sigue siendo manejable. El problema aparece con imágenes que se están utilizando hoy en día, con una cantidad de pixeles muchísima más alta, donde el coste computacional se dispara.

En una Red de Neuronas Convolucional se utilizan tres tipos principales de capas para formar una red de este tipo: *Convolutional Layer*, *Pooling Layer* y *Fully Connected Layer*.

- **Convolutional Layer** (Capa Convolucional): Procesa la salida de neuronas que están conectadas únicamente a una parte de la imagen.
- **Pooling Layer** (Capa de agrupamiento): Procesa la información para quedarse con la parte más importante y reducir el tamaño de los datos.
- **Fully Connected Layer** (Capa completamente conectada): Se emplea en las últimas etapas para realizar la clasificación.

Este tipo de arquitecturas fueron implementadas por primera vez en la época de los 80 inspiradas en la observación de estructuras naturales en el córtex de algunos animales [17]. No fue hasta 2006 cuando se comenzó a utilizar estas redes por medio de Unidades de Procesamiento Gráfico (GPUs) [18]. A partir de 2010, se produjo un aumento continuo de los recursos computacionales, permitiendo utilizar redes con muchas más capas y pesos, comenzando así el comienzo de un exponencial progreso del reconocimiento y verificación facial.

A lo largo de la última década, por primera vez una red de neuronas había logrado unos resultados similares a los de un ser humano, ganando en 2011 más de una competición de reconocimiento facial. El *ImageNet Large Scale Visual Recognition Challenge* es un *benchmark* de clasificación de imágenes muy considerado y positivamente valorado para comparar resultados de modelos de reconocimiento de imágenes [19]. Entre las redes convolucionales (CNNs) que han destacado en este concurso destacan Alexnet, quedando en el primer puesto en 2012, donde la utilización de GPUs dio un giro completo y redujo en más de un diez por ciento

la tasa de error anterior. A partir de ese momento, las CNNs comenzaron a tener una enorme importancia en estas competiciones, reduciendo constantemente la tasa de error gracias al diseño de nuevas arquitecturas. A partir de 2016 esa tasa de error se situaba por debajo del 5% en la mayoría de los modelos presentados.

3.2.4. REDES SIAMESAS

Una red siamesa consiste en replicar parte de la arquitectura de una red de neuronas, para posteriormente fusionarse en una o varias capas comunes, que permiten obtener resultados a través de todo el proceso previo de ambas replicas. Con ello se permite comparar dos inputs, en nuestro caso dos imágenes de personas, extraer las "características" de cada uno de los inputs, y realizar cualquier tipo de método (clasificador) que defina un output fácilmente interpretado como resultado, en este caso un *afirmativo* o *negativo* en función de si la persona es correctamente verificada.

Los primeros modelos siameses para la verificación facial surgieron a comienzos de este siglo, donde utilizaron métodos de reducción de dimensiones para posteriormente comparar las características entre pares de imágenes como en el caso de Chopra [20].

Más tarde aparecieron nuevos modelos de Deep Learning, utilizando nuevas formas de extraer características de los usuarios. Sin embargo, la estructura siempre ha sido la misma y no ha variado en su forma básica: una parte simétrica e independiente, junto con una comparación entre ambas.

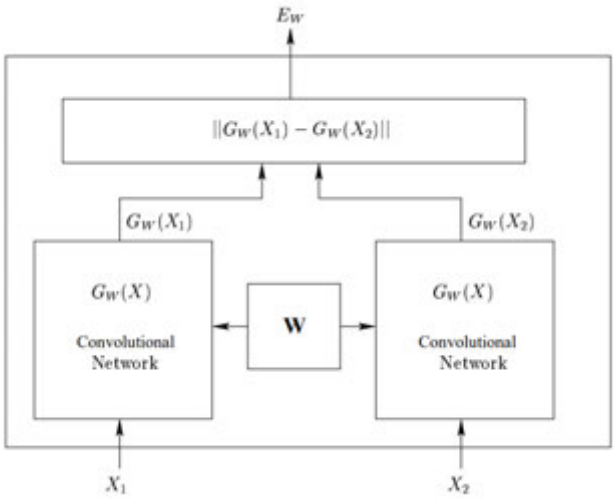


Ilustración 5: Arquitectura de Red Siamesa [20].

Más adelante en el capítulo de desarrollo estará definido y desarrollado el diseño de la red siamesa utilizada en este trabajo.

3.2.5. DATASETS

Para el entrenamiento del modelo de clasificación, se ha realizado una previa búsqueda de los datos necesarios que se deben utilizar como inputs o entradas al modelo.

El conjunto de datos recogidos, organizados y preparados para ser utilizados en las fases de entrenamiento y evaluación se denomina *dataset* o conjunto de datos. Los datasets son una parte fundamental para este tipo de estudios; por ello, es necesario analizarlos e interpretarlos antes de pasar al diseño del modelo. La salida del modelo va a depender directamente de la entrada con la que se ha trabajado. Un modelo con inputs mediocres es muy difícil de lograr unos resultados robustos.

Durante las primeras etapas de este estudio, se utilizaron pequeños datasets como el famoso MNIST [21]. Se basa en un conjunto de números escritos a mano del 0 al 9, que han permitido comprender el concepto de clasificación a través de redes de neuronas convolucionales.

ImageNet [22] es un dataset de un gran tamaño, con más de 14 millones de imágenes, muy utilizado y apoyado las investigaciones de visión artificial. Se trata de un dataset con imágenes de distintos entornos y elementos, como animales, paisajes, objetos, etc. Se ha utilizado para con continuo proceso previo de aprendizaje antes de realizar este estudio. Al tratarse de imágenes muy variadas y ajeno a rostros de personas, se ha descartado este conjunto de datos para la realización del mismo.

El siguiente dataset analizado ha sido el llamado PubFig [23]. Es un dataset formado exclusivamente por imágenes con rostros de personas obtenidas en internet. Contiene concretamente 58797 imágenes de 200 personas.

Labeled Faces in the Wild (LWF) [24] es otro dataset que contiene exclusivamente imágenes con rostros de personas de personas famosas obtenidas a través de Internet. En este caso contiene 13233 imágenes de 5749 personas distintas, donde 1680 personas tienen más de una imagen. Este conjunto de datos contiene menos imágenes en comparación con PubFig, pero el número de personas, y por ende de clases totales a comparar, es mucho mayor. Un total de 200 personas no se ha considerado como un número aceptable para poder realizar un modelo sobre verificación facial.

Finalmente, LFW ha sido el dataset escogido para este estudio, al tratarse de un conjunto de datos público y de naturaleza académica exclusivamente. Sin embargo, existen ciertas limitaciones a tener en cuenta sobre este conjunto de datos, entre ellas:

- Algunos grupos de personas no están correctamente identificados en LFW. Grupos como bebés no están incluidos, o grupos como personas mayores de 80 años o niños pequeños están incluidos en muy poca proporción. Además, existe una leve desproporción entre hombres y mujeres, además de una desproporción entre personas de distinta procedencia.
- No están reflejadas todas las condiciones desfavorables en cuanto a la calidad de imágenes. Escasa luminosidad, posicionamiento de rostros en los extremos de la imagen, casos con muy poca resolución, etc.

3.3. RESULTADOS DE MODELOS AJENOS AL TRABAJO

Para poder realizar una evaluación completa del modelo, los resultados se han analizado y comparado con diversas publicaciones existentes a nivel internacional sobre la verificación facial.

La comparativa de estos modelos publicados anteriormente se ha obtenido a partir de las matrices ROC, al tratarse de una medida muy utilizada en todas ellas, y que su vez permite representar tanto la tasa de falsos positivos como indirectamente la tasa de falsos negativos gracias a su eje vertical "Ratio de Verdaderos Positivos". Cuando mayor sea el valor, menor será esta última tasa de falsos negativos.

Se han tenido en cuenta varios aspectos en cuanto a los resultados del modelo y los de publicaciones ajenas:

- Los modelos entrenados en el presente trabajo han utilizado solamente una parte del dataset de "LFW" para posteriormente evaluar estos modelos con el resto de imágenes disjuntas, definido y explicado con anterioridad. Por tanto, las comparaciones entre modelos no pueden interpretadas de manera directa.
- En comparación con modelos entrenados en trabajos ajenos, estos han sido entrenados mediante la utilización de datasets de entrenamiento mucho mayores a LFW como pueden ser *CelebFaces* [25] , utilizando en ocasiones la combinación de varios datasets para poder realizar estos entrenamientos.
- Las evaluaciones de los modelos del presente trabajo han sido realizadas con la parte complementaria del dataset de LFW utilizado en el entrenamiento, como se explica en anteriores capítulos. En modelos ajenos, sin embargo, estos han sido evaluados con el completo dataset de LFW.

3.3.1.1. MODELOS TRADICIONALES

A continuación, se muestran algunos resultados de modelos tradicionales, sin la aplicación de técnicas de aprendizaje profundo:

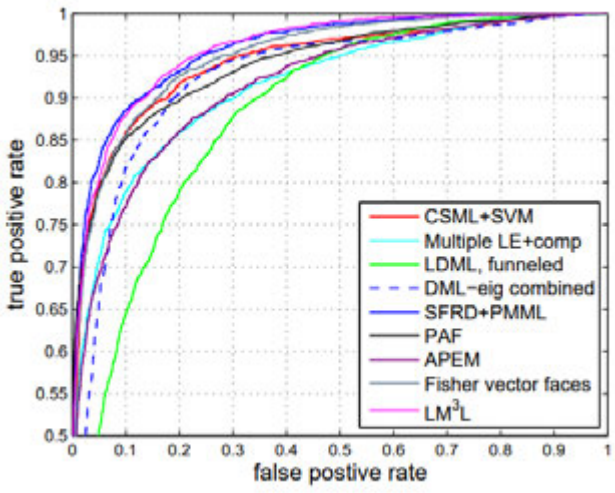


Ilustración 6: Comparativa de curvas ROC entre técnicas tradicionales [26].

3.3.1.2. EVALUACIÓN SER HUMANO

A través de las pruebas realizadas en 2009 se elaboró una curva ROC basada en las evaluaciones de verificación facial realizadas por personas sobre el dataset de LFW.

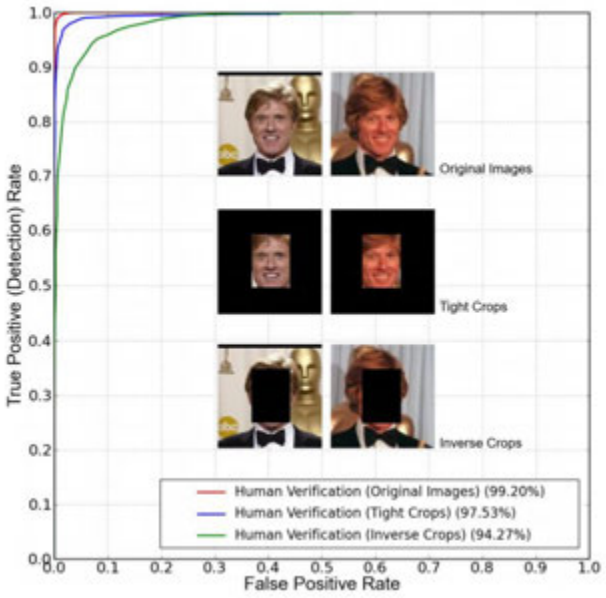


Ilustración 7: resultados de LFW evaluados por seres humanos con imágenes originales , recortadas, e inversamente recordadas ocultando su rostro [12].

3.3.1.3. MODELOS CON APRENDIZAJE PROFUNDO

Se han obtenido varios modelos de aprendizaje profundo o Deep Learning con unos resultados muy positivos. Entre ellos destacan los de la publicación de Sun, Wang y Tank [27]que a través de un entrenamiento de su modelo basado en el dataset de más de 100000 imágenes de CelebFaces+B se pudo lograr una precisión en LFW superior al 99%. A partir de ese momento, las redes implementadas mostraban unos resultados casi perfectos.

Se muestran a continuación la curva ROC de algunas de las redes más conocidas utilizando modelos con mayor complejidad y con el uso de Deep Learning:

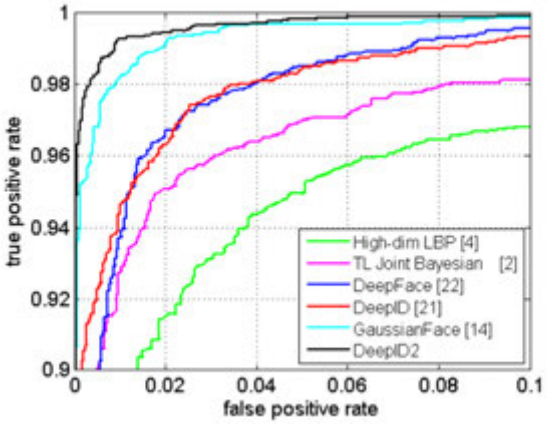


Ilustración 8: Comparativa de curvas ROC entre técnicas de Deep Learning [27].

Para poder comparar los tiempos de inferencia, aspecto muy importante en este trabajo cuyo fin es obtener los mejores resultados en el menor tiempo posible, se han analizado algunas de las redes más potentes. DeepId2 se basa en la extracción previa de varios trozos o "patches" de la misma imagen para posteriormente realizar tantas inferencias como *patches* se obtengan para lograr un resultado común a todas ellas. La exactitud y el rendimiento obtenido por cada imagen en función del número de *patches* utilizado es:

# patches	1	2	4	8	16	25
accuracy (%)	95.43	97.28	97.75	98.55	98.93	98.97
time (ms)	1.7	3.4	6.1	11	23	35

Ilustración 9: Exactitud y tiempos de extracción de características por cada imagen en el modelo DeepId2, en función del número de trozos o "patches" utilizados. Ejecutado en tarjeta gráfica Titan [27].

Los resultados parecen ser bastante positivos tanto en la exactitud como en la velocidad de ejecución, sin embargo, es necesario considerar que estos tiempos han sido los obtenidos por cada imagen, y no por cada par de imagen, además de su posterior clasificación de cada par. Además, el hardware utilizado ha sido puntero en comparación con otros resultados, al utilizar recursos de procesamiento gráfico de "Nvidia Titan".

Otros resultados recogidos han sido resultado de la conferencia de biometría facial de Shenzhen de 2017 [28], donde se obtuvieron resultados desde un 94% hasta 98% de precisión aproximadamente:

Method	Networks	Accuracy	Protocol
LBPNet [17]	1	94.04%	Unsupervised
DeepFace [5]	1	95.92%	Unsupervised
DeepID2 [6]	1	95.43%	Unsupervised
Webface [11]	1	96.13%	Unsupervised
Webface + PCA [11]	1	96.30%	Unsupervised
Light CNN A [14]	1	97.77%	Unsupervised
Light CNN B [14]	1	98.13%	Unsupervised
Improved Light CNN	**1**	**96.80%**	**Unsupervised**

Ilustración 10: Exactitud de los modelos evaluados por la 12ª conferencia de biometría de Shenzhen [11].

Los tiempos de inferencia por cada una de las redes evaluadas están incluidos en la tabla proporcionada. Se ha utilizado un microprocesador Intel Xeon e3 para estos resultados:

Model	Parameters	Feature dimension	Times (per image)	Storage space
DeepFace	120,000K	4096	198 ms	-
VGG [10]	27,749K	4096	455 ms	553 MB
CenterLoss [18]	19,596K	1024	140 ms	-
Light CNN A	3,961K	256	68 ms	26 MB
Light CNN B	5,556K	256	65 ms	32.8 MB
Improved Light CNN	2,165K	256	29 ms	8.7 MB

Ilustración 11: Número de parámetros, dimensiones de características, tiempos de inferencia y espacio utilizado de disco de cada modelo evaluado por la 12ª conferencia de biometría de Shenzhen [28].

Se puede ver cómo en este caso las velocidades de ejecución han sido menores, al tratarse de hardware CPU en lugar de procesamiento gráfico.

Desde 2014 se han publicado diversos modelos que han permitido un aumento en cuanto a los resultados de distintos datasets, no solo en el utilizado en este estudio [29]. En ellos, se ha desarrollado un continuo aumento cronológico de las métricas utilizadas a medida que se proponen nuevos resultados:

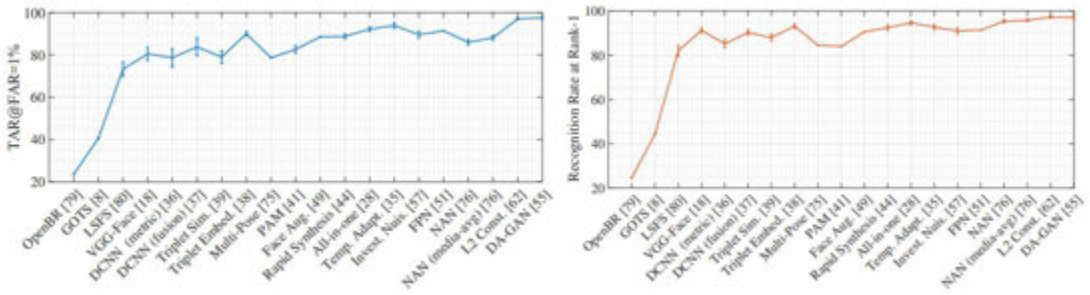


Ilustración 12: Progreso en el dataset de evaluación de IJB-A sobre dos métricas: Ratio de FP o Falsas Alarmas (FAR) y Exactitud de los Top-1 en clasificación [29].

Todos ellos han sido implementados utilizando distintas técnicas, pero ninguno de ellos ha utilizado el diseño de las redes siamesas, y menos a través de la técnica de maximización de la perdida por falsos positivos durante el entrenamiento, al tratarse de modelos utilizados para clasificación, y por tanto no especializados en la verificación biométrica.

3.3.1.4. CONCLUSIONES MODELOS AJENOS

Comparando estos modelos con los propios de este trabajo, claramente las curvas ROC de los modelos ajenos obtienen muy buenos resultados en cuanto a la eficacia de los mismos, con curvas que alcanzan un alto ratio de verdaderos positivos y al mismo tiempo un bajo ratio de falsos positivos (la curva se acerca más a la esquina superior izquierda). Estos modelos no han sido entrenados con el mismo número de datos que han sido entrenados el modelo de este trabajo, por lo que es importante recordar que solamente puede resultar una aproximación a los resultados que se obtienen en el modelo propio.

Sin embargo, este resultado de la curva ROC no solamente se tiene en cuenta de forma individual, sino que además es necesario considerar los tiempos de predicción o verificación de cada modelo. Como se ha podido ver, estos tiempos obtenidos en algunos de los resultados pueden dar una idea de la situación en la que se encuentran, teniendo en cuenta cada equipo de hardware utilizado, mucho más potente en estos en comparación con un dispositivo móvil, por lo que estos tiempos serán inferiores en el caso de utilizar estos últimos.

4. DESARROLLO

En este capítulo se podrá comprender el diseño y funcionamiento del modelo de red siamesa que se ha logrado implementar, junto con sus respectivas alternativas de arquitecturas de redes de neuronas convolucionales que posteriormente se han podido evaluar para una mejor comprensión de las mismas.

4.1. ARQUITECTURAS DE LAS REDES IMPLEMENTADAS

En capítulos anteriores se ha definido el concepto de red neuronal siamesa. Durante el desarrollo, se han ido analizando múltiples arquitecturas en función del objetivo a alcanzar, hasta dar con algunas ellas que permiten obtener grandes resultados potencialmente positivos para la posterior aplicación de ellas en el conjunto del modelo.

Todo el proyecto ha sido implementado en el lenguaje de programación *Python,* utilizando librerías de código abierto muy destacadas en la comunidad de Machine Learning, entre ellas:

- **Tensorflow**: es una librería base desarrollada por Google, que permite implementar modelos de Deep Learning de bajo nivel mediante grafos, aunque también permite facilitar la implementación de los mismos gracias a otras librerías de más alto nivel. Para la elaboración del código implementado se ha utilizado Tensorflow versión 1, que permite el desarrollo de modelos a un bajo nivel en comparación con la versión TF2.0.
- **Keras**: es otra librería que facilita enormemente la implementación de redes neuronales, así como el entrenamiento y las pruebas a realizar (*testing*).
- **OpenVC** y **Imutils**: librerías de visión artificial que han sido utilizadas en este proyecto para procesar las imágenes, recortarlas, rotarlas y alinearlas previamente para el entrenamiento y testeo de la red, así como la utilización posterior.
- **Numpy**: librería fundamental utilizada para algebra linear en Python.
- **Tensorboard**: es una librería de Tensorflow que permite visualizar grafos y todo tipo de información de manera gráfica. Muy necesaria para la obtención de resultados en este proyecto.

Para la mejor comprensión del diseño de la red siamesa, se ha decidido realizar una división en cuanto la naturaleza de la red en dos bloques:

- **Bloque 1**: Bloque de extracción de características.
- **Bloque 2**: Bloque de clasificación.

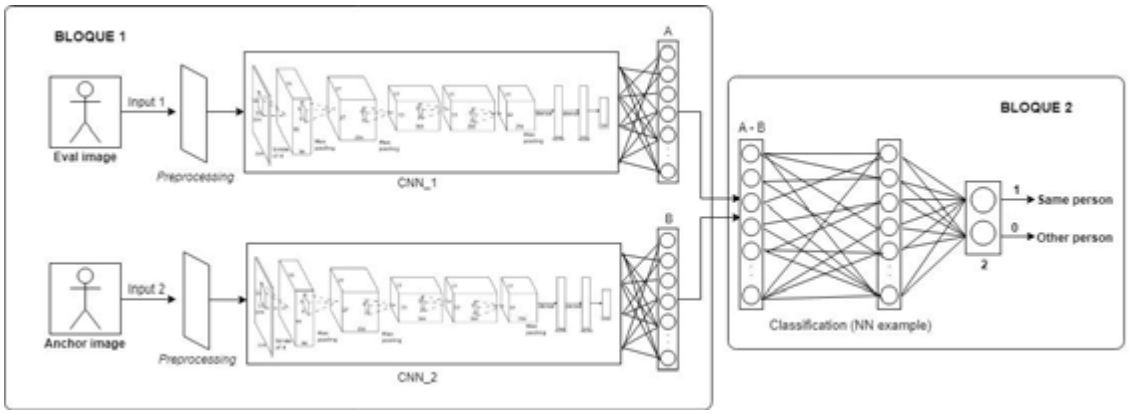


Ilustración 13. Arquitectura general de la Red Neuronal Convolucional Siamesa planteada y utilizada.

Cada bloque realiza una función distinta: el bloque 1 consiste en dos flujos de datos completamente iguales y paralelos entre sí, los cuales procesan una imagen cada uno y extraen la mayor información de cada una de las imágenes para una posterior comparación; por otro lado, el bloque 2 compara las dos salidas del bloque 1 para obtener los resultados finales.

A continuación, se explica cada uno de estos bloques, definiendo detalladamente la estructura y flujo de los datos.

4.1.1. BLOQUE I: EXTRACCIÓN DE CARACTERÍSTICAS

La extracción de características es una fase fundamental a la hora de diseñar una red siamesa. En ella se realizan una serie de inferencias a través de las capas convolucionales, hasta obtener una capa final llamada cuello de botella o *bottleneck*, utilizada posteriormente como input del bloque II.

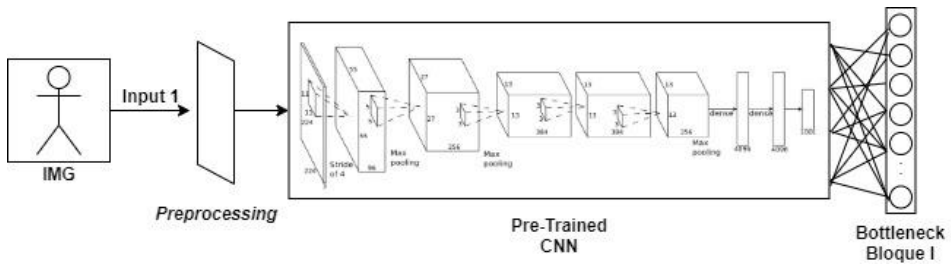


Ilustración 14: Diseño Arquitectura Bloque I.

Este *bottleneck* permite recoger cada una de las características recogidas por las capas convolucionales en cada una de las imágenes, que posteriormente se usarán para comparar y obtener el resultado. El objetivo a alcanzar por la red consiste en extraer la mayor información sobre las imágenes para que posteriormente puedan ser clasificadas de la mejor manera posible.

El bloque de extracción de características es el más grande y pesado de los dos. Para obtener los distintos bottlenecks, las imágenes de entrada al bloque serán transformadas a través de capas de aprendizaje profundo que hemos visto anteriormente (convolucionales, *max-pooling*, etc.). Estas capas son más ligeras en comparación con capas *fully-connected*, pero para la extracción de características se necesitan muchas de ellas hasta poder obtener el cuello de botella o

bottleneck, provocando así grandes esfuerzos en diseñar y crear una arquitectura que permita obtener los mejores resultados.

Para la implementación y diseño de este bloque, se ha realizado un análisis de las redes más destacadas hoy en día, creadas por organizaciones con gran capacidad para explorar y crear aquellas que sean las más óptimas para este tipo de problema de extracción biométrica. Para ello, es necesario primeramente entender los conceptos básicos de los que van a tratar cada una de ellas:

- ***Overfitting* o sobreajuste:** es el efecto de sobreentrenar un modelo de Machine Learning, llegando al punto de reducir la precisión real en la fase de evaluación. No es capaz de predecir correctamente el resultado en otros casos a partir de lo aprendido con los datos de entrenamiento.
- **Suavizado de capa (*label smoothing*)** [30]: es una técnica de regularización la cual distorsiona las etiquetas o *labels* (Y) para evitar que la red aprenda de una manera estricta las etiquetas de entrenamiento, mejorando la robustez y eficiencia de los resultados, reduciendo así el *overfitting*. Suele utilizarse en etiquetas tipo "one hot encoding"

- **Normalización de lotes (*batch normalization*)**: es una solución que se aplica en forma de un pre-procesamiento de los inputs en una capa de la red. Evita que cada capa deba aprender a adaptarse a una nueva distribución distinta en cada paso del entrenamiento por cada input (problema llamado *Internal Covariance Shift*). Se aplica una normalización sobre los mismos antes de ser procesado por la capa en cuestión.

- ***One hot encoding***: es una técnica de etiquetado de clases optimizada para algoritmos de machine Learning, que transforma variables categóricas en numéricas. En otras palabras, transforma cada una de las clases o etiquetas en vectores con un rango numérico, donde la posición es la que determina la clase, y el valor en sí determina el grado o intensidad que debe tener cada una de las posiciones. El *one-hot* cumple siempre que destaque una posición frente al resto dependiendo de la clase o categoría en cuestión.

- **Stochastic Gradient Descent (descenso de gradientes estocástico)**: es un optimizador que permite ir actualizando los parámetros del modelo sin tener que procesar todas las muestras de datos del entrenamiento, sino que permite actualizarse en pequeños subgrupos. Suelen ser más rápidos en converger a los valores óptimos que el descenso de gradiente tradicional.

4.1.1.1. INCEPTION-RESNET-V1 E INCEPTION-RESNET-V2

Estas arquitecturas están formadas por una combinación de dos redes de neuronales utilizadas en el pasado: Inception [31], arquitectura creada por Google, y la red de aprendizaje con residuos ResNet, desarrollada por Microsoft [32]. Cada una de ellas ha sido modificada progresivamente, lanzando distintas versiones de ellas.

Las arquitecturas Inception-ResNet poseen un magnífico rendimiento en resultados de previos estudios [33], y sobre todo es conocida por una gran velocidad de aprendizaje y convergencia en la fase de entrenamiento así como a la hora de realizar inferencias en las imágenes, permitiendo muy buenos resultados en menos tiempo y sin necesidad de altos costes computacionales [34].

La arquitectura está basada en las redes Inception y ResNet. Ambas se componen de una estructura bastante diferente, pero la combinación de ambas permite obtener muchos aspectos positivos de cada una de ellas.

INCEPTION

En el reconocimiento de imágenes por medio de redes de neuronas artificiales, es muy posible que las imágenes tanto para el entrenamiento como para el testeo y predicción no dispongan de una composición equivalente en cada una de ellas. El tamaño del *kernel* que se debe de especificar en el diseño de la red es muy propenso a variar entre cada una ellas. Para solucionar dicho problema, anteriormente se necesitaban redes con más cantidad de capas, aumentando la profundidad total de la red, dando lugar a mayores costes computacionales que empeoraban los resultados con respecto al tiempo de ejecución, y sobretodo dificultaba el control del *overfitting*.

Con la red de Inception [33], se consiguen unos resultados muy parecidos a las redes anteriores, solo que la profundidad disminuye drásticamente gracias al aumento de amplitud o anchura de cada capa.

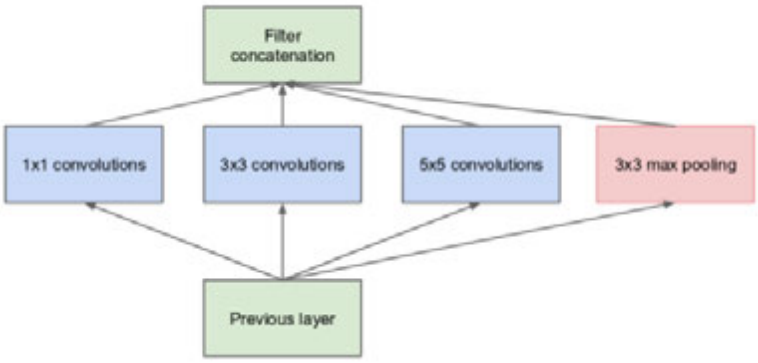


Ilustración 15 Módulo Inception v1 [33].

Como se puede apreciar en la imagen posterior, la red se compone de *módulos* que se adaptan a la imagen específica para utilizar el tamaño de *kernel* más apropiado en cada una de ellas. Con ello además se consigue evitar que se reduzca la "anchura" de la red, con el fin de evitar perdida de información como se produce en las redes más profundas y "finas". Estos *módulos* han ido perfeccionándose poco a poco en base a este tipo de arquitectura, apareciendo versiones como la Inception v2, v3 y v4.

Estas últimas versiones se basan siempre en la misma estructura de módulos o bloques como en Inception-V1: uno o varios tipos de *kernels*, junto con la capa de *MaxPooling*:

En **Inception-V2** se optimizan los pesos de los *kernels* más grandes, sustituyendo un *kernel* grande por varios pequeños de forma secuencial. Además, se añade *batch normalization* [35] o normalización de bloques.

En **Inception-V3** se aplica un nuevo optimizador (RMSProp Optimizer) que mejora el aprendizaje de la red en menores pasos [33]. Además, se añaden convoluciones con *kernels* de 7x7 factorizadas, es decir, compuestas por pequeñas convoluciones de 1x3 y 3x1 que disminuyen el número de parámetros obteniendo los mismos resultados finales. Se añade además normalización de lotes por cada uno de los clasificadores auxiliares añadidos previamente en las otras arquitecturas, y se añade un suavizador de capas o *label smoothing,* utilizado para evitar *overfitting*.

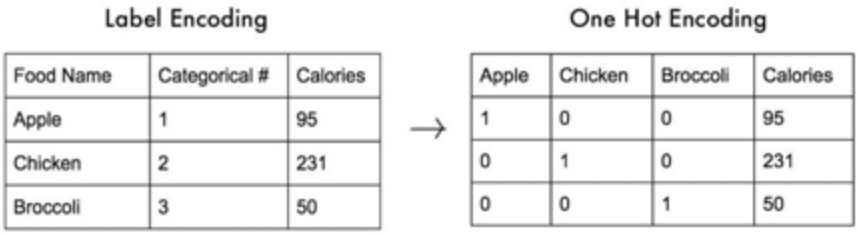


Ilustración 16: Diferencia entre label y one hot encoding [36].

En **Inception-V4**, se modifican los módulos para que sean más uniformes, incluido el bloque "*stem*". Además, se añaden uno bloques especiales llamados "*reduction blocks".*

RESNET

Las redes profundas son más complicadas de optimizar, por ello, con esta red se ha podido lograr que se puedan utilizar muchas más capas sin perder ningún error de optimización. Esto es gracias a unos módulos llamados bloques residuales, que permiten en la fase de propagación hacia atrás o "***back-propagation"*** que se propague el error desde las etapas finales a las iniciales con mayor facilidad. Destacan dos ramas paralelas en la red distribuidas en bloques:

1. Una rama tradicional, con cada uno de los parámetros y pesos.
2. Una rama conectada directamente con el input, llamado bloque identidad o "*identity block".*

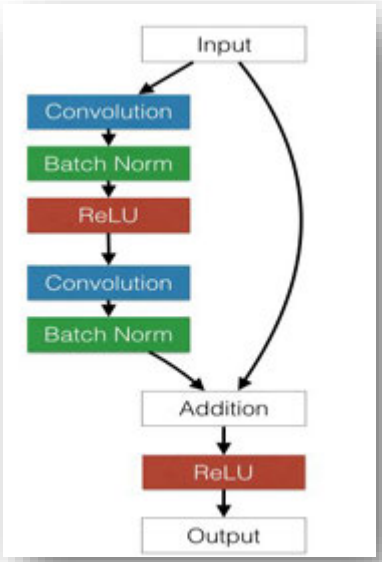


Ilustración 17: Arquitectura de un bloque residual de ResNet [31].

Estos bloques parten de una red lineal tradicional, solo que se incluye una función residual entre la capa input del bloque y la salida, que nos permite averiguar qué necesitamos añadir o quitar del input para poder seguir con la siguiente capa

En la ResNet, los bloques residuales están formados por 2 capas convolucionales 3x3, colocados uno detrás de otro. Solamente se utiliza una capa *Fully Connected* al final de la red, para dar la salida de 1000 clases.

ARQUITECTURAS FINALES

Con la combinación de ambas arquitecturas, se ha podido lograr una fusión de las mejores características que ofrecen cada una de ellas: la amplitud o anchura de las redes de Inception, que evitan la perdida de información, junto con la gran profundidad de permiten las redes con bloques residuales manteniendo el control del *overfitting*.

Se han utilizado en el proyecto ambos modelos Inception-ResNetV1 de Inception-ResNetV2:

- El primero, **Inception-ResNetV1**, está formado por la misma estructura que la arquitectura de InceptionV3, con la diferencia de que a ésta se han agregado las características de una red residual ResNet, permitiendo acelerar el entrenamiento gracias a la mayor propagación hacia atrás de la modificación de los pesos por cada paso.
- El segundo, **Inception-ResNetV2**, posee la misma estructura que la arquitectura de InceptionV4, con la diferencia de que se han añadido, al igual que la versión V1, las características de una red residual ResNet.

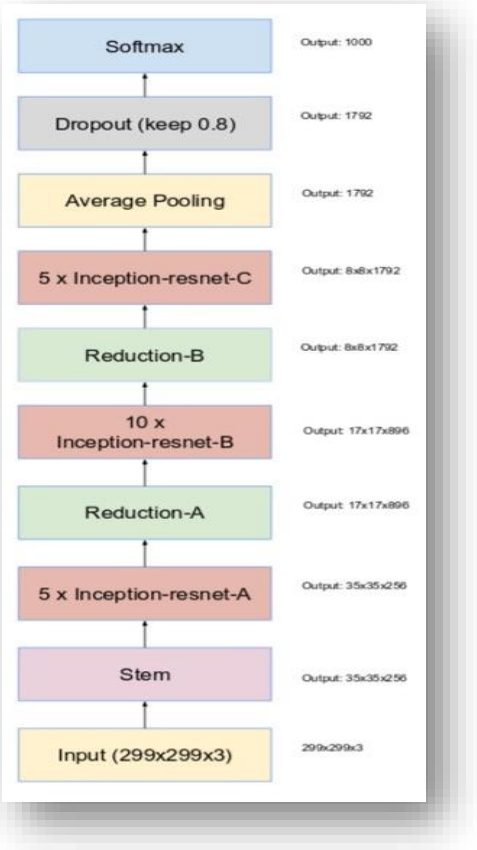


Ilustración 18: Arquitectura final de Inception-ResNet v1 y v2. Bloque de obtención de características [37].

4.1.1.2. MOBILENETV1

A lo largo de los últimos años han surgido arquitecturas muy especializadas para el procesamiento de imágenes, siendo algunas de ellas muy destacables en los resultados obtenidos al utilizarlas en dispositivos ligeros y con capacidad computacional limitada, como pueden ser los dispositivos móviles. Entre ellas destacan las arquitecturas de MobileNet. La primera de ellas fue la versión V1 [38], con una velocidad tres veces mayor a la arquitectura Inception, pudiendo utilizarse en *smartphones* modernos a 20 fotogramas por segundo (FPS) con resultados del 70% de exactitud utilizando *ImageNet* [39].

La red utiliza un nuevo concepto de arquitectura, donde algunos bloques están basados en un tipo de convolución diferente, llamado "*Deepwise*" y "*Pointwise*".

La convolución de tipo "Deepwise" (DW Conv) realiza cada una de ellas de una manera independiente por cada canal, como pueden ser los colores Rojo Verde o Azul (RGB) en el caso concreto de este estudio. Por tanto, cada uno de estos canales tendrá su propia convolución de manera separada, a diferencia de las convoluciones tradicionales donde todos ellos acababan transformándose en un solo *output*.

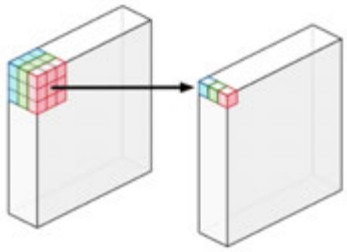


Ilustración 19: Convolución Deepwise.

La convolución *PointWise* es totalmente lo contrario a *DeepWise*. Los inputs en forma de canales se analizan todos a la vez, solo que en este caso el "kernel" o núcleo de convolución es solo de tamaño 1. Es decir, por cada convolución se utiliza un solo valor de cada canal para obtener un solo valor como output.

Un bloque de MobileNet tiene la siguiente estructura:

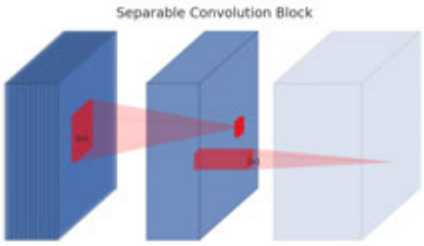


Ilustración 20: Bloque de MobileNetV1.

Se utiliza una primera transformación con convolución *DeepWise*, y otra de *PointWise* a continuación.

La arquitectura global de MobileNetV1 es la siguiente:

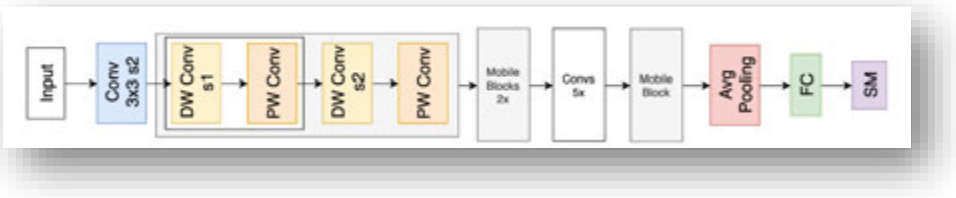


Ilustración 21: Arquitectura de MobileNetV1

4.1.1.3. MOBILENETV2

En abril de 2018 se publicó una nueva versión de *MobileNet* que mejoraba a su antecesora [40]. En ella se incluyen las *conexiones residuales,* misma técnica que las redes *ResNet, y cuellos de botella* por cada bloque. Con las conexiones residuales se consigue ir modificando durante el entrenamiento los pesos de las capas más profundas o lejanas, evitando que se queden inactivas o "muertas". Con los cuellos de botella, se logra evitar la pérdida de información ocasionada por

las no-linealidades entre cada capa. Se ha demostrado previamente que el uso excesivo de capas no lineales entre si (los inputs y outputs entre capas tienen dimensiones distintas) provocan esta pérdida de información [41].

La estructura de un bloque de MobileNetV2 es la siguiente:

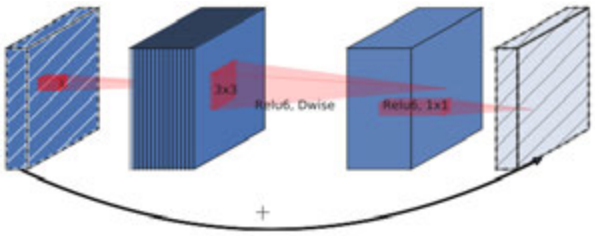


Ilustración 22: Bloque de MobileNetV2 [40].

A diferencia de su antecesora, este diseño se incluye la conexión residual y una transformación final que termina formando una capa de una sola dimensión (bottleneck).

Network	Top 1	Params	MAdds	CPU
MobileNetV1	70.6	4.2M	575M	113ms
ShuffleNet (1.5)	71.5	**3.4M**	292M	-
ShuffleNet (x2)	73.7	5.4M	524M	-
NasNet-A	74.0	5.3M	564M	183ms
MobileNetV2	**72.0**	**3.4M**	**300M**	**75ms**
MobileNetV2 (1.4)	**74.7**	6.9M	585M	**143ms**

Tabla 1: Comparativa Top1, total parámetros, MAdds y tiempo de inferencia de redes MobileNet frente a parecidas del momento [35].

4.1.1.4. MOBILENETV3

A finales de 2019 un grupo de investigadores desarrollaron una nueva versión de MobileNet. Permite utilizar las CPUs de los dispositivos móviles directamente, a un rendimiento considerablemente mayor que su versión anterior, MobileNetV2. Permite aumentar la exactitud en ImageNet en comparación con sus redes antecesoras, además de reducir su latencia y ahorra de batería en dispositivos móviles.

El bloque de MobileNetV3 viene dado por:

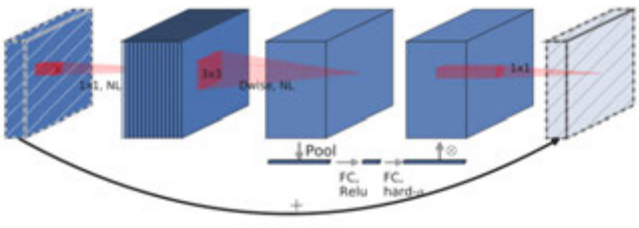


Ilustración 23: Bloque de MobileNetV3 [42].

Se trata de una combinación de las tecnologías de *MobileNetV2* y *Squeeze* and Excite, o Exprimir o Excitar [43]. Esta última está diseñada para mejorar los resultados que se obtienen a través de la utilización de capas adicionales en puntos estratégicos de la arquitectura de la red, sin llegar a alterarla en su forma. En este caso se añaden estas capas a continuación de la convolución de *Deepwise*, manteniendo las dimensiones y tamaño entre ellas.

4.1.1.5. ANÁLISIS PREVIO DE ARQUITECTURAS

A continuación, se ha realizado un análisis de las arquitecturas anteriormente explicadas para poder escoger aquellas que permitan una mejora de los resultados y permitan entender con mayor profundidad el comportamiento de éstas. Se han obtenido resultados de pruebas realizadas por distintos autores.

Los parámetros que se han considerado para realizar este análisis han sido los siguientes:

- **Exactitud** ("*accuracy*"). Mide proporción de la cantidad de aciertos en predicciones, tanto positivos como negativos, que coinciden con los resultados reales. En numerosas publicaciones se ha considerado la exactitud a través de dos métricas: el top1 y top5 de exactitud.
- **Complejidad del modelo**: se determina a través del número total de parámetros que tiene cada modelo, medido en MB. Es importante esta métrica ya que permite conocer el tamaño total necesario en memoria para que el modelo pueda ejecutarse correctamente.
- **Tiempo de inferencia**. Uno de las métricas más importantes, indica el tiempo necesario para que el modelo pueda procesar los inputs y devolver el output. Muy interesante considerarla en este caso de modelos en sistemas con limitación de recursos, ya que un tiempo de inferencia mayor obtiene un resultado global bastante inferior que tiempos de inferencia menores.
- **Coste computacional**: corresponde al número total de operaciones computacionales que el modelo debe realizar, normalmente medido en Operaciones de Coma Flotante por Segundo o *FLOPs* ("*Floating Point Operations per Second*"). Está métrica está bastante correlacionada con el tiempo de inferencia.
- **Memoria total utilizada**. No solo el tamaño del modelo sino todo el resto de recursos, como pueden ser aquellos que se necesitan para cada lote.

INCEPTION-RESNET-V1

La red de Inception-ResNet es una combinación de dos tipos de redes que aprovechan las ventajas de cada una de ellas, como se ha explicado en anteriores apartados. Logra mantener una exactitud muy elevada sin alterar la complejidad del modelo. Según un equipo formado de Google Inc. [44], los resultados que han obtenido tras realizar múltiples evaluaciones han sido muy parecidos a los obtenidos por la red de Inception-V3, como se puede ver en la imagen a continuación. La evaluación ha sido realizada a través del método de "top-5 error", un método de evaluación muy utilizado en modelos de clasificación, mostrando el error obtenido en cada paso del entrenamiento sobre los resultados de las predicciones. En esta evaluación se considera

como “error” cuando la clase a predecir o acertar (label real) no se encuentra entre las 5 clases obtenidas en los resultados de la predicción con mayor similitud.

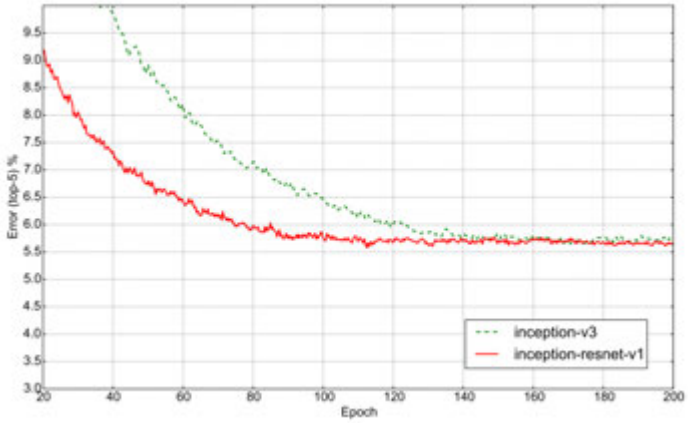


Ilustración 24: top-5 error en la fase de entrenamiento de Inception-V3 e Inception-ResNet-V1 [44].

Los resultados muestran perfectamente cómo la red de Inception-ResNet-V1 converge al valor óptimo con mayor rapidez y en un número menor de pasos (“*Epochs*”), debido al aprovechamiento de las redes residuales de las redes ResNet.

INCEPTION-RESNET-V2

Al igual que su primera versión, la red Inception-ResNet está basada en el mismo concepto de redes *Inception* junto con la adición de características de las redes residuales. En este caso la red está basada en Inception-V4, utilizando mismos bloques convolucionales que ésta y añadiendo las capas residuales entre ellas. Con ello, se obtienen resultados similares a su antecesora, como se puede ver en la imagen obtenida a gracias al equipo de Google:

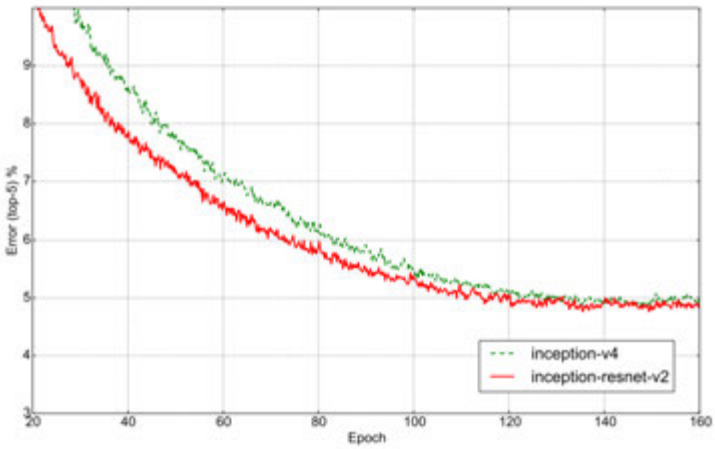


Ilustración 25: : top-5 error en la fase de entrenamiento de Inception-V4 e Inception-ResNet-V2 [44].

La red de Inception-ResNet-V2 converge ligeramente antes de la red Inception-V4, gracias de nuevo a la implementación de la tecnología de ResNet.

Finalmente, se puede ver en la siguiente tabla cómo las redes de Inception-ResNet superan los resultados de las redes Inception, obtenida también a través de investigaciones del equipo de Google Inc [44]. La columna de "*Top-1 Error*" muestra, en un modelo de clasificación, la proporción de fallos de predicción que se obtienen cuando el valor o clase real no coincide con el valor de predicción o clase más acertada por el modelo.

Network	Top-1 Error	Top-5 Error
BN-Inception [6]	25.2%	7.8%
Inception-v3 [15]	21.2%	5.6%
Inception-ResNet-v1	21.3%	5.5%
Inception-v4	20.0%	5.0%
Inception-ResNet-v2	19.9%	4.9%

Tabla 2: resultados de evaluación de las redes Inception e Inception-ResNet [44].

Se concluye con esto que las redes de Inception-ResNet poseen una mayor exactitud sin alterar la complejidad del modelo, con la ventaja además del aumento en la velocidad de entrenamiento.

MOBILENET-V2

Con la llegada de las redes MobileNet, la complejidad del modelo y el tiempo de inferencia se reducen drásticamente en comparación con las redes Inception-ResNet, a cambio de reducir la exactitud en las evaluaciones. En el siguiente gráfico se muestra las principales arquitecturas de reconocimiento facial utilizadas hasta 2018, con los modelos de MobileNet-V1, MobileNet-V2 e Inception-Resnet-V2 evaluados, además de Inception-V3 e Incpetion-V4.

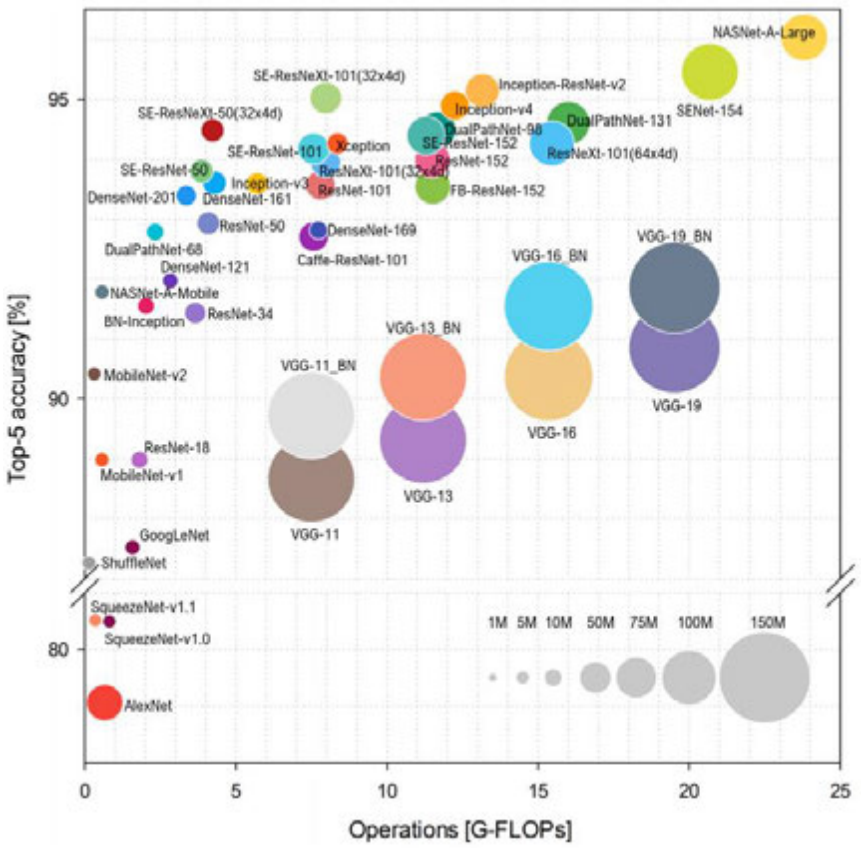


Ilustración 26: resultados de distintos modelos sobre la exactitud de tipo Top-5 (eje vertical), la complejidad computacional (eje horizontal) y complejidad del modelo (tamaño de cada círculo) [45].

Se puede ver claramente cómo las redes MobileNet tienen una complejidad computacional menor a 1, equivalente a más de 10-15 veces menos la complejidad computacional de las redes de Inception-ResNet. La exactitud de estas últimas sin embargo es superior.

Además, especialmente la red de MobileNetV2 logra una mayor exactitud de más del 90% con una complejidad computacional de menos de medio G-FLOP, superando en todos los aspectos a su antecesora MobileNet-V1.

MOBILENETV3

Se ha logrado un aumento de la exactitud en resultados de evaluación en distintas publicaciones, manteniendo prácticamente constante la latencia. El equipo de Google publicó a finales de 2019 los siguientes gráficos donde se muestra esta mejora [46]:

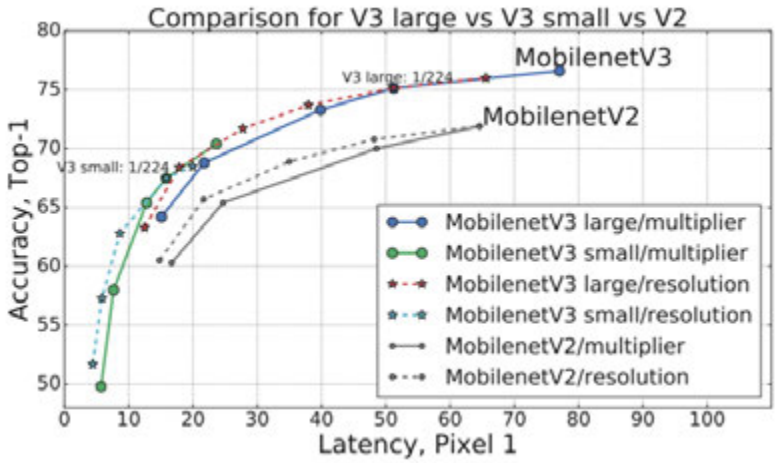


Ilustración 27: Resultados de varias redes utilizadas en TFLite de MobileNet-V2 y V3. Relación de exactitud (Top-1) y latencia. [46]

Esto se ha debido gracias a la reducción de la complejidad del modelo de MobileNetV3 con respecto a su predecesora manteniendo la exactitud, permitiendo ser más ligera y rápida en mismos procesamientos.

4.1.1.6. COMPARATIVA DE LAS ARQUITECTURAS DEL ESTUDIO.

Tras el análisis de las arquitecturas que han tenido en cuenta para el estudio, se ha elaborado una tabla comparativa con todos los resultados recogidos de las distintas fuentes oficiales de cada una de las publicaciones de cada modelo. De esta manera se puede comprobar de manera clara el potencial a primera vista de cada una de ellas, antes de realizar la implementación de las mismas.

Arquitectura	Complejidad del modelo (Millones de parámetros)	Exactitud / Accuracy (Top-1) (%)	Exactitud/ Accuracy (Top-5) (%)	Tiempo de Inferencia (32 lotes)(ms)	Coste Computacional (G-FLOPs)	Memoria Utilizada
Inception-V3	27.1	78.8	94.4	1.89	5.72	830
Inception-ResNet-V1	-	78.8	94.5	-	6	-
Inception-V4	46	80.1	94.8	3.77	12.3	900
Inception-ResNet-V2	55.8	80.1	95.1	4.88	13.3	950
MobileNet-V2	6	74.9	92.5	0.786	0.03	700
MobileNet-V3	5.4	75.2	-	0.659	-	-

Tabla 3: comparativa de resultados de los modelos obtenidos por distintas fuentes.

Claramente se observa que no ha sido posible recoger todos los datos de la tabla, debido a la gran falta de información

veraz acerca de los resultados de cada una de ellas, y la imposibilidad de poder comparar varias arquitecturas en entornos completamente distintos (hardware distinto, hiperparámetros, etc.).

Observando los datos recogidos, se ha concluido que existe un aumento considerable en la exactitud de los modelos de Inception e Inception-ResNet en comparación con las redes MobileNet. Sin embargo, estas últimas han mejorado de manera mucho más diferenciadora importantes resultados de rendimiento como la complejidad del modelo, el tiempo de inferencia, y el coste computacional. Para el enfoque del estudio, relacionado con la implementación del modelo en dispositivos con ciertas limitaciones, estos tres aspectos son muy importantes y considerados a la hora de realizar conclusiones.

Por otro lado, dado los resultados de las redes de Inception, y debido a la gran similitud de sus versiones "mejoradas" de Inception-ResNet, finalmente se han dado como descartadas en la implementación de las mismas.

Por tanto, las arquitecturas finales implementadas para este estudio han sido:

- Inception-ResNet-V1
- Inpection-ResNet-V2
- MobileNet-V2
- MobileNet-V3

4.1.2. BLOQUE II: CLASIFICACIÓN

Una vez analizadas y seleccionadas las arquitecturas más destacables para ser utilizadas en el bloque I, se ha procedido a desarrollar segundo y último bloque.

El bloque de clasificación es el encargado de realizar las operaciones necesarias para determinar si en dos imágenes distintas aparece la misma persona o no. En el Aprendizaje Automático o *Machine Learning*, existen tres tipos o categorías de modelos según su implementación [47]:

- **Supervisado**: los algoritmos utilizan datos que están "etiquetados". Estas etiquetas o *labels* se utilizan para "enseñar" al algoritmo lo que debe de aprender, de manera que cuando este recibe un dato de entrada él mismo debe de predecir y acertar en el resultado basándose en estas etiquetas. Se aplica en problemas de clasificación como este proyecto y en problemas de regresión.
- **No supervisado:** no se disponen de datos "etiquetados", solo de datos de entrada, por tanto, tienen un carácter exploratorio. Se aplica en algoritmos de clústering sobre todo, además de en Análisis de Componentes principales (*Independent Component Analysis*) y descomposición en valores singulares (*Singular Value Decomposition*).
- **Aprendizaje de refuerzo**: se basa en el aprendizaje propio a base de sucesos en el entorno por medio de un proceso de retroalimentación. Es un método de ensayo-error.

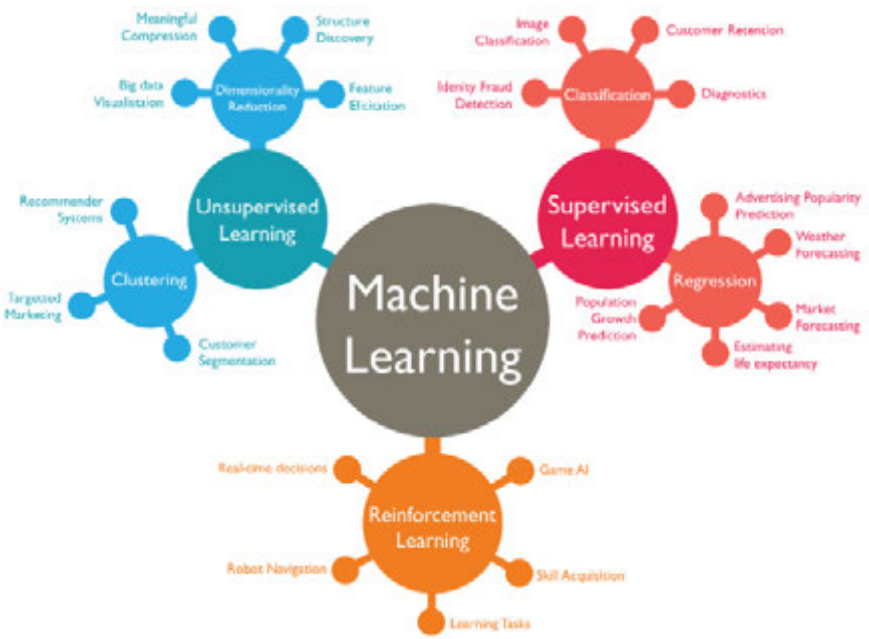


Ilustración 28: Aprendizaje supervisado, no supervisado y de refuerzo. Fuente: StepUp Analytics [48].

El tipo de aprendizaje para este proyecto, basado en redes de neuronas, ha sido el de clasificación binaria, incluido en el grupo de aprendizaje supervisado. Por tanto, para este entrenamiento, es necesario una serie de etiquetas o *labels* que permitan al algoritmo aprender a distinguir entre imágenes de una misma persona o personas distintas.

La estructura básica estará formada por:

- Input: "*array*" o lista con la información de las características, obtenida mediante una diferencia absoluta de las salidas del bloque 1 o *bottlenecks.*
- Output: "*array*" o lista de dos posiciones, una es grado de similitud que existe entre ambas imágenes de ser la misma persona, y otra es el grado de no similitud. Positivos y negativos. Se trata de una clasificación binaria.

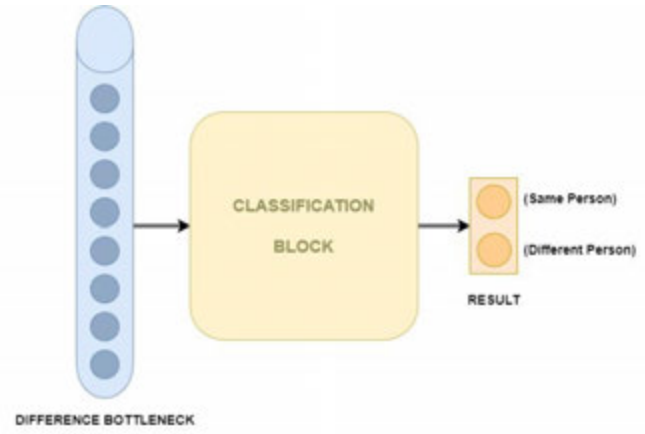


Ilustración 29: Bloque del Bloque II (Clasificación). Fuente: elaboración propia.

La estructura de este bloque siempre será la misma, con un input adaptado a la salida del bloque I y mismo tipo de output. Tras la realización de algunas pruebas utilizando distintas capas y tamaño total de la red, este bloque de clasificación se compone de dos capas "Fully Connected": una de 512 neuronas o conexiones, y otra de 2.

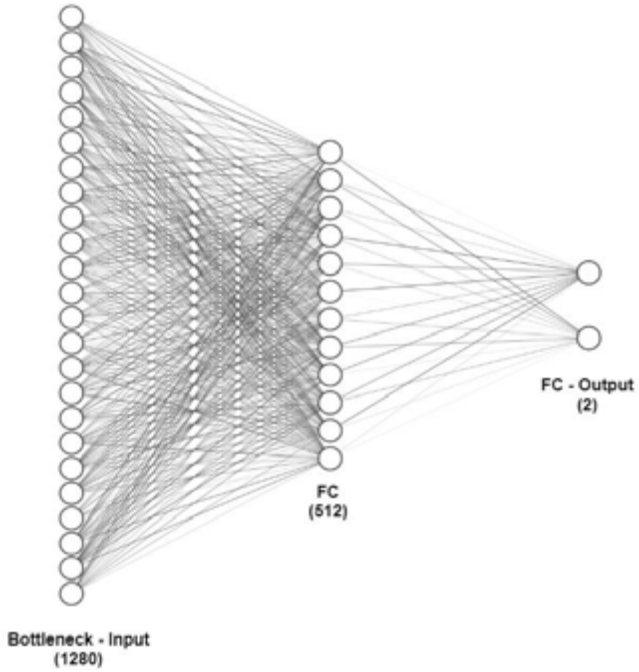


Ilustración 30: Arquitectura del bloque II (Clasificación), alto nivel. Fuente: elaboración propia.

Como se observa en la anterior imagen, la red es relativamente sencilla, con el fin de optimizar la velocidad de los resultados y de reducir el sobre-entrenamiento, que se explica con más detalle en los siguientes apartados.

En cuanto a su arquitectura en un bajo nivel, este bloque se compone de la siguiente estructura:

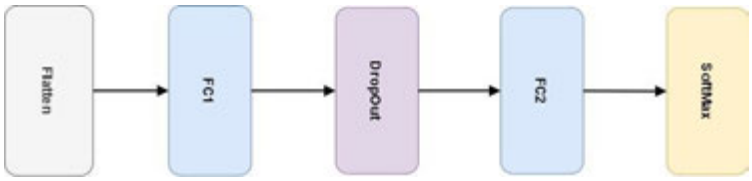


Ilustración 31: Estructura de la red de neuronas del bloque II, bajo nivel.

Como puede observarse, la primera capa llamada **"*flatten*"** consiste en "aplanar" cada una de las dimensiones que puedan existir a una sola dimensión. Con ello, la red se asegura la inexistencia de ningún tipo de jerarquía entre los datos de entrada, tratando cada uno de ellos de igual manera. Las dimensiones varían según los datos de entrada, que vienen determinados por las salidas de cada una de las arquitecturas del bloque I. Varían entre 1280 y 1792 valores.

A continuación, se ha añadido una capa **"*Fully-Connected (FC)*"** con 512 neuronas. Dependiendo del tamaño del input, el número de parámetros que tiene esta capa será de (n + 1) * 512, donde

n indica el número de valores de entrada (tamaño de la capa *flatten*) y "1" es el *bias* o sesgo que se incluye por cada salida de esta capa.

Se ha incluido una capa de **"*DropOut*"**, donde lo único que hace es desactivar una pequeña proporción de las neuronas de la previa capa, con el fin de obligar a que "aprendan" de nuevo, evitando de esta forma que el modelo se sobre-entrene sobre los datos de entrenamiento, provocando el llamado "*overfitting*".

Después se ha añadido otra capa más **"*Fully-Connected (FC)*"** de solamente dos neuronas o salidas. Son las que nos van a indicar el resultado final, siendo una de ellas el grado en que se trata de la misma persona y otra el grado en que se trata de personas distintas. El número de parámetros serán de 1024, obtenido tras multiplicar las 512 neuronas de la capa anterior con las 2 nuevas.

Es importante recordar que la arquitectura utilizada en base a redes neuronales ha sido aquella que ha podido asegurar unos resultados de entrenamiento muy robustos, pudiendo haberse utilizado otros tipos de clasificadores distintos como *Support Vector Machines (SVM*), método de los k vecinos más cercanos *(k-Nearest Neighbor*, KNN), Arboles de decisión o Clasificadores Bayesianos [49].

4.1.2.1. GRAFO TENSORFLOW

Para la implementación de la red neuronal se ha utilizado la librería de Tensorflow, que ofrece además un módulo que permite obtener información gráfica, *Tensorboard*. Con él se ha podido analizar la estructura del flujo de los datos que transcurren por la red:

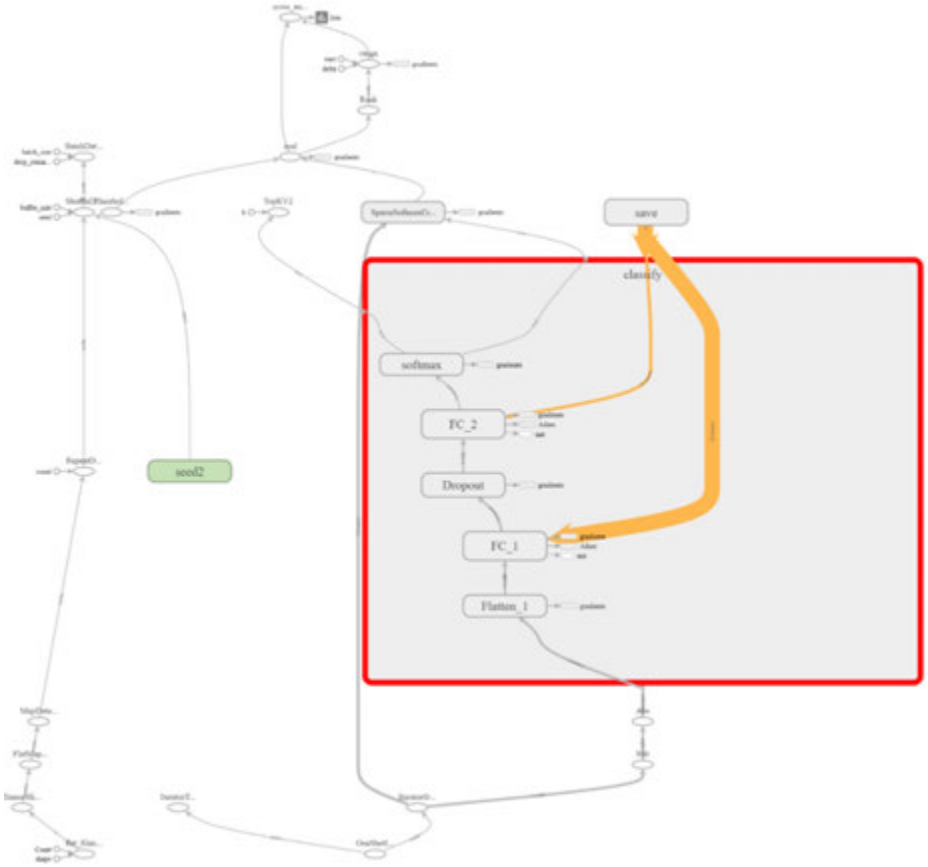


Ilustración 32: Grafo de Tensorflow de la red neuronal del bloque II.

El cuadro con marco rojo consiste en el propio clasificador junto con sus capas: *Flatten*, FC1, *Dropout*, FC2 y *Softmax*. Es importante destacar los datos de entrada y salida en el clasificador.

En cuanto a los datos de entrada, se incluyen las propias imágenes en forma de "iteradores", y "*Save*" son los pesos entrenados que se cargan cada vez que inicialicemos la red para realizar inferencias.

En cuanto los datos de salida, se incluyen por un lado los resultados obtenidos por medio de la función TopKV2 para averiguar qué clase es la más acertada (0, distintas personas, o 1, misma persona); y por otro lado se incluye la pérdida o "*loss*" de la predicción del modelo, linealmente dependiente de la semilla utilizada, necesaria para obtener los resultados de los distintos modelos.

4.1.2.2. SALIDAS

EL modelo de este bloque se trata de un problema de predicción de clases binario, donde los resultados se etiquetan como positivos, misma persona, o negativos, personas distintas.

El resultado u output de este modelo de clasificación binario consiste en dos valores: uno que nos indica a qué clase pertenecen las imágenes, uno o cero, dependiendo de si se tratan de la misma persona o no (positiva y negativa); y otro valor que nos indica el grado de similitud, de 0

a 1, utilizado para contrastarlo con el umbral de discriminación. Este umbral determina el grado de similitud que es necesario para considerar ambas imágenes o inputs como mismas personas: cuanto mayor es este umbral (cercano a 1), mayor similitud debe existir entre ambos rostros de las imágenes para que se consideren como una sola persona, y viceversa.

Por tanto, es posible que el modelo identifique como positivo un resultado (los inputs pertenecen a la misma persona) pero, sin embargo, el umbral de similitud es mayor al grado de similitud del resultado, considerando por tanto el resultado como negativo. En la fase de entrenamiento y evaluación tras la implementación, este umbral de similitud se ha ido alterando controladamente para poder contrastar cada uno de los resultados, que serán reflejados en las siguientes páginas.

En total, existen cuatro posibles resultados, como en cualquier problema de predicción de clases binario:

- **Verdadero Positivo** (VP) *o True Positive* (TP): si el resultado de la predicción es positivo y el valor dado también es positivo.
- **Verdadero Negativo** (VN) o *True Negative* (TN): si el resultado de la predicción es negativo y el valor dado también es negativo.
- **Falso Positivo** (FP) o False Positive: si el resultado de la predicción es positivo pero el resultado realmente no lo es. También llamado error de tipo I.
- **Falso Negativo** (FN) o *False Negative*: cuando el resultado de la predicción es negativo pero el valor realmente es positivo.

Estos cuatro resultados se han interpretado formando una Tabla de contingencia o Matriz de confusión:

		Valor en la realidad		
		p	***n***	**total**
Predicción outcome	***p'***	Verdaderos Positivos	Falsos Positivos	P'
	n'	Falsos Negativos	Verdaderos Negativos	N'
	total	P	N	

Ilustración 33: Matriz de confusión.

En cuanto a los errores, un error de Tipo II o de Falsos Negativos (FN) consiste en identificar como personas distintas al par de imágenes que realmente son la misma persona. Sin embargo, en el caso concreto de este estudio, se considera como error más crítico e imprescindible evitar al error de tipo I o Falsos Positivos (FP). En concreto, el error consiste en identificar como mismas personas aquellas que realmente no son, y por ello, se debe evitar a toda costa, ya que las futuras aplicaciones del modelo pueden ser utilizadas para desbloqueos de terminales para posterior uso de datos personal, entre otras. Más adelante se muestran de una manera detallada los resultados y conclusiones acerca de la matriz de confusión y métricas relacionadas

como la Exactitud del modelo, la Precisión, la Sensibilidad, la Especificad, la razón de Falsos Positivos y Negativos, y la ratio de F1.

4.2. PREPROCESAMIENTO

El procesamiento es aquel conjunto de operaciones y modificaciones previas que se realizan en los datos que van ser utilizados por la red neuronal, con el fin de estandarizar estos datos y entrenar la red de la mejor manera posible.

4.2.1. NORMALIZACIÓN DE LOS DATOS

Para ello, en el dataset del proyecto, todas las imágenes han sido modificadas y preparadas previamente utilizando librerías de Python. Una de ellas ha sido MTCNN [11]. El procesado que han tenido estas imágenes ha sido de un alineamiento y recorte en cuanto a los rostros, preparando el modelo para que pueda focalizar sus "esfuerzos" en la distinción entre personas, y no en el alineamiento de ellas simultáneamente.

Las imágenes por tanto han sido alineadas y recortadas en función del rostro de las personas, formando un nuevo dataset "normalizado", que se utilizará para la fase de entrenamiento y evaluación del modelo.

4.2.2. FILTRADO DE IMÁGENES VÁLIDAS PARA EL ENTRENAMIENTO

Una vez normalizadas las imágenes del dataset, lo siguiente que se ha realizado ha sido un filtrado de imágenes en función de varios parámetros. Con ello lo que se pretende es obtener un dataset final con mismo número de pares de imágenes de casos positivos (misma persona) como de casos negativos (personas distintas). Los parámetros son:

- **Número mínimo de imágenes por clase o persona** para el dataset de entrenamiento. Se ha establecido una cantidad de al menos 15 imágenes por clase o persona en el dataset, y en caso contrario se descartan las imágenes de esa clase. De esta manera, el bloque de clasificación binaria será capaz de poder aprender correctamente y de manera más equitativa los casos positivos donde cada par de imágenes pertenezca a la misma persona, al disponer de más de 15 imágenes que la identifiquen. Para las clases con menos de 15 imágenes, todas ellas han formado el dataset de evaluación.
- **Número máximo de imágenes por clase o persona** para el dataset de entrenamiento. Se ha establecido un límite por arriba sobre el número máximo de imágenes en el dataset que puedan pertenecer realmente a una misma persona, concretamente de 75. Con ello obtenemos la mayor cantidad posible de información sin que pueda llegar a ser demasiado desigual la calidad de datos entre clases o personas.

De esta manera, se dispone de 96 clases o personas distintas en el dataset de entrenamiento, con más de 15 y menos de 75 imágenes cada una de ellas; y más de 5000 clases entre 1 y 14 imágenes en cada una de ellas para el dataset de evaluación, la mayoría de ellas con una sola imagen.

4.2.3. AUMENTACIÓN DE DATOS O "DATA AUGMENTATION"

Para el entrenamiento de verificación facial, es altamente necesario realizar un pre-procesado de las imágenes de entrada, a través de la aumentación de datos.

Este concepto consiste propiamente en aumentar la cantidad de datos que disponemos para trabajar en el modelo utilizando un mismo dataset. Para ello, se utilizan las imágenes del dataset para poder realizar réplicas de ellas con pequeñas modificaciones, de manera que por cada imagen que se dispone, se pueden añadir nuevas imágenes sobre una misma, aumentando de esa manera el tamaño total del dataset.

Para ello, se han generado 10 imágenes aleatorias por cada imagen seleccionada del dataset, realizando transformaciones sobre la imagen base utilizando rotaciones aleatorias, desplazamientos aleatorios horizontales y verticales, exposición aleatoria, distorsiones ligeras, ampliación o zoom del tamaño de la imagen aleatoria (pequeña o grande), o volteo con eje horizontal.

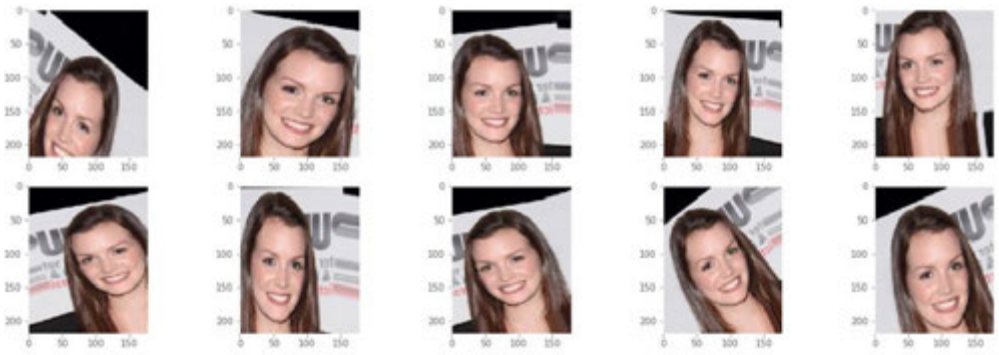


Ilustración 34: Ejemplos extremos de aumentación de datos [50].

En este momento, se dispone de dos datasets, uno de entrenamiento y otro de evaluación, formados por imágenes normalizadas y alineadas, y ordenadas en directorios con clases.

4.2.4. OBTENCIÓN DE LOS "BOTTLENECKS" DEL BLOQUE I

Una vez preparado el dataset antes del bloque I, se ha procedido a realizar las inferencias de cada una de las imágenes tanto del dataset de entrenamiento como del dataset de evaluación. En otras palabras, se han extraído las características de cada una de ellas por medio del primer bloque del modelo, obteniendo un vector final llamado cuello de botella o bottleneck, tal y como se explica en anteriores apartados.

Con ello, lo que se logra es evitar tener que realizar este paso en futuros procedimientos en cuanto a la optimización de parámetros, al no variar en absoluto las salidas de estas inferencias.

4.2.5. PREPARACIÓN DEL DATASET DE ENTRENAMIENTO DEL BLOQUE II

A partir del resultado del pre-procesamiento, se ha diseñado una nueva estructura de carga de datos para el bloque II, formado por tres elementos, donde:

- Bottleneck de imagen 1: resultado obtenido tras realizar la inferencia en el bloque I de la primera imagen a comparar.
- Bottleneck de imagen 2: resultado obtenido tras realizar la inferencia en el bloque I de la segunda imagen a comparar.
- Etiqueta o *label*: número entero que puede ser 1 (cuando ambas imágenes pertenecen a la misma persona) o 0 (cuando ambas imágenes no pertenecen a la misma persona).

Cabe destacar en esta fase previa al entrenamiento que los bottlenecks siguen cargándose a partir del propio dataset de los mismos, donde esta nueva estructura de datos solamente se indica la dirección de cada uno de ellos.

Con este nuevo conjunto de datos, se ha calculado el valor de entrada final que utiliza el clasificador. Consiste en un nuevo vector, obtenido por el valor absoluto de la diferencia de los valores de cada par de bottlenecks obtenidos. Este nuevo vector mantendrá el mismo tamaño y forma que los bottlenecks de las imágenes.

$$\overrightarrow{\text{DifferenceBottleneck}} = \begin{pmatrix} |a - a'| \\ |b - b'| \\ |c - c'| \\ \vdots \\ |n - n'| \end{pmatrix}$$

Ilustración 35: Input de entrada de clasificador. Bloque II.

$$\overrightarrow{\text{bottleneck1}} = \begin{pmatrix} a \\ b \\ c \\ \vdots \\ n \end{pmatrix} \qquad \overrightarrow{\text{bottleneck2}} = \begin{pmatrix} a' \\ b' \\ c' \\ \vdots \\ n' \end{pmatrix}$$

Ilustración 36: Bottlenecks de salida del bloque I.

Con este nuevo valor se ha procedido a realizar la fase de entrenamiento para cada uno de los modelos, calculando dicho valor cada vez que se realicen inferencias con el bloque II.

4.3. ENTRENAMIENTO:

La fase de entrenamiento ha sido realizada en el Bloque II de la arquitectura del modelo. En ella se ha establecido un nuevo dataset que permite clasificar de forma binaria como se ha mencionado anteriormente.

Como input, se ha utilizado el cuello de botella obtenido tras obtener los valores absolutos de las diferencias entre cada uno de las imágenes procesadas por el bloque I, siendo "M" el tamaño del Bottleneck, y N el tamaño del dataset (número total de inputs):

$DifferenceBottleneck_1$ [M]
$DifferenceBottleneck_2$ [M]
$DifferenceBottleneck_3$ [M]
…
$DifferenceBottleneck_N$ [M]

Tabla 4: Input del Bloque II.

Y como output, se obtiene el valor binario de 0 o 1 que indica si se trata de la misma persona o no, junto con su grado de intensidad o de similitud en esa misma clase:

Descripción	1. Id Clase	2. Grado de similitud
Misma persona	1	$x \in \{0,1\}$
Personas distintas	0	$y \in \{0,1\}$

Tabla 5: Output del Bloque II.

El grado de similitud es un valor real normalizado y comprendido entre 0 y 1 que indica qué clase es la que corresponde en cada caso: el grado con mayor valor positivo entre estas dos clases ("0" y "1") será la decisión final para la predicción del modelo en cada inferencia.

No obstante, se ha establecido un parámetro, comprendido entre 0 y 1, que permite establecer un umbral de aceptación para poder considerar dos imágenes como una misma persona. En otras palabras, si en la predicción se obtiene que el resultado es la misma persona (Id de Clase igual a "1"), y dicho umbral es menos que el grado de similitud de la misma clase (misma persona), la predicción se considera como imágenes de la misma persona. Pero, en el caso donde la predicción considera que el resultado es la misma persona (Id de Clase igual a "1"), pero el umbral de similitud es mayor al grado de similitud de esa misma clase (misma persona), entonces el resultado final de la predicción se considera como imágenes de distintas personas.

4.3.1. LA TÉCNICA DE AUMENTO DE PERDIDA POR FALSOS POSITIVOS

Además, como técnica realizada propia del trabajo y utilizada durante el entrenamiento, se ha establecido una política de las pérdidas o "*loss*" en cada etapa del entrenamiento. Consiste en aplicar una tasa multiplicadora adicional de la pérdida en cada *batch* o paso, para aquellos casos donde exista una inferencia con al menos un falso positivo. De esta manera, el modelo podrá aprender de manera más estricta a evitar este tipo de casos con mayor importancia que el resto.

4.3.2. HÍPER-PARÁMETROS Y ARGUMENTOS DEL ENTRENAMIENTO

A lo largo del estudio se han realizado numerosos procesos de entrenamiento, tratando de combinar los principales valores que pueden afectar al resultado. En *Machine Learning, l*os híper-parámetros son aquellos parámetros que se transfieren a los distintos modelos. Los híper-parámetros y argumentos que se han utilizado para realizar las combinaciones de los modelos son:

- **Modelo Bloque** I: anteriormente analizados, son los modelos implementados para la extracción de características. Se han utilizado:
    - Inception-ResNet-V1
    - Inception-ResNet-V2
    - MobileNet-V2
    - MobileNet-V3
- **Semilla** (*semilla*): número entero que permita una consistencia en todas las ejecuciones, ya que las librerías de la implementación realizan fragmentaciones de datos aleatorios en cada ejecución. Con las semillas se ha podido controlar esta aleatoriedad. Se han utilizado cuatro semillas aleatorias para realizar tres análisis distintos y obtener una mayor precisión de los resultados de los modelos (13, 25, 29 y 51)
- **Tamaño del Lote** (*batch_size*): número de elementos, imágenes en este caso, que se utilizan por cada iteración cada vez que se actualizan los pesos del modelo. Se han utilizado tamaños del lote de 8, 16 y 32 imágenes.
- **Numero de pasos** (*max_steps*): número de pasos o iteraciones que se realizan durante la fase de entrenamiento. Se han probado con 250, 500, 1000 y 2000 iteraciones.
- **Dropout**: proporción de neuronas que se desactivan por cada iteración. Se ha utilizado en Tensorfow el DropOutKeepProp, que indica la proporción de neuronas que no se desactivan o realizan alguna modificación. Se ha mantenido en 0.85 (85%).
- **Tasa de Aprendizaje** (*learning_rate*): proporción o grado de magnitud que determina el tamaño del paso en cada iteración necesario para optimizar la función de pérdida o "*loss".* En otras palabras, es la velocidad a la que un modelo de aprendizaje automático "aprende". Se han utilizado los valores 0.01, 0.001, 0.0001.

En total, teniendo en cuenta cada una de las posibilidades existen por cada hiperparámetro u argumento, por medio de combinatoria se han realizado un total de 4x4x3x4x1x3 = 576 entrenamientos distintos.

Dichos entrenamientos han podido realizarse perfectamente en un ordenador portátil personal ("*Xiaomi Mi Laptop Pro 15,6 inch Intel Core i7-10510U NVIDIA GeForce MX250 16GB DDR4 RAM 1TB SSD"*). La razón de esta posibilidad de entrenamiento en local ha sido gracias a la previa

obtención de los bottleneck de cada imagen en el bloque I, que han sido obtenidos una sola vez por cada modelo, realizando únicamente:

- $\boldsymbol{Total\ inferencias\ de\ entrenamiento\ Bloque\ I} =$
  $Nº\ de\ Arquitecturas\ Bloque\ I * Nº\ total\ de\ imágenes\ de\ entrenamiento =$
  $= 4 * 2828 = \mathbf{11312}\ inferencias$

Únicamente 12312 inferencias en total para la fase de entrenamiento, que se han utilizado para realizar el entrenamiento de las 576 instancias de modelos distintas. De no haber realizado estas inferencias del bloque I previas al entrenamiento, se tendrían que haber realizado un total de:

- $\boldsymbol{Total\ inferencias\ de\ entrenamiento\ Bloque\ I\ (caso\ ineficiente)} =$
  $Nº\ de\ Modelos\ Totales * Nº\ túplas\ entrenamiento\ Bloque\ II\ por\ modelo *$
  $Nº\ Imágenes\ por\ túpla\ Bloque\ II\ =$
  $= 576 * 5760 * 2 = \mathbf{6635520}\ inferencias\ totales\ de\ Bloque\ I$

6635520 inferencias en esta fase, una estrategia ineficiente de más de x586 veces el tiempo y procesamiento necesario de no haber realizado esta estrategia de separación por bloques del modelo.

4.3.3. REDUCIENDO OVERFITTING

El overfitting o sobre-entrenamiento es uno de los principales obstáculos en este tipo de problemas. Es posible que los resultados realizados solamente con datos de entrenamiento sean muy positivos, pero a la hora de realizar pruebas con datos que no se incluyen en este dataset sean bastante negativos en comparación. Esto se debe al fenómeno de *sobre-entrenamiento u "overfitting"*, que se produce debido al extremo ajuste de las predicciones en los datos de entrenamiento, que, a la hora de generalizar el modelo, éste ha aprendido pequeñas características específicas que resultan irrelevantes para nuevos datos.

Para evitar este problema:

- **Aumentar los datos** de entrenamiento todo lo posible. Es imprescindible y directamente proporcional utilizar más datos en el entrenamiento del modelo para mejorar los resultados, y además evitar el *overfitting,* siempre que estos nuevos datos sean correctos y preparados para ser utilizados en el modelo. "*Data Augmentation"* es una clara opción para añadir nuevos datos adicionales, como se ha utilizado en este estudio.
- **Optimizando híper-parámetros**: eligiendo los argumentos necesarios para la fase de entrenamiento se consigue optimizar el modelo de una mejor forma. Existen numerosas formas de optimizar los híper-parámetros, entre ellas destaca por ejemplo crear un dataset de validación, que consiste en utilizar un pequeño conjunto de datos no utilizados para entrenar el modelo, que permita detectar este *overfitting* durante de fase de entrenamiento y detenerlo en el número de pasos más óptimo. Esta técnica se denomina "*early-stoping*".

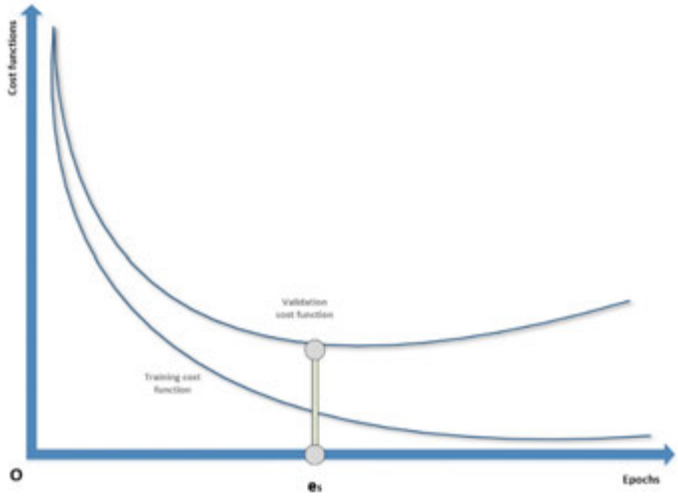


Ilustración 37: "Early Stoping" en una fase de entrenamiento [51].

En el caso de este estudio no se ha utilizado un dataset de validación, se ha decidido utilizar el mayor número de datos de entrenamiento posibles, y por ello se han realizado distintas ejecuciones variando el numero máximos de pasos de entrenamiento ("*max-steps*").

- **Añadir regularización**. Es un término que consiste en la suavización de los resultados, evitando que el modelo se sobreajuste a los datos de entrenamiento. Entre sus técnicas destaca:
    - **DropOut**. Como se ha mencionado previamente, el "*DropOut"* consiste en la "desactivación" de aleatoria de neuronas en función de una proporción de desactivación aleatoria. Esta técnica se ha añadido en el bloque II como se observa en el apartado de la arquitectura, además de la implementación en las distintas arquitecturas del bloque I.
    - Normalización **por lotes ("*batch normalization*")** [35]**.** Es una técnica que consiste en añadir un paso extra entre la red, que permite normalizar las salidas de cada neurona por separado. Con ello se reducen las oscilaciones de la función de coste de modelo, pudiendo aumentar la tasa de aprendizaje y por tanto evitando que la red termine en un mínimo local provocado por los datos de entrenamiento. Se ha utilizado esta técnica en las arquitecturas del bloque I.
    - **Regularización L1** [52]: a la función de coste del modelo, se le añade un nuevo término a la ecuación, que permite suavizar el entrenamiento, evitando que pueda ser sobre-entrenado al exceso ajuste de los datos de entrenamiento. En concreto, se añade un parámetro (lambda) que multiplica a la suma de los valores absolutos de los "*w"* pesos en el modelo (siendo los peos $\hat{y} = wx + b$).

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} |w_i|$$

Ilustración 38: Función de coste con Regularización L1 [52].

Siendo “y” el valor real e “ŷ” el valor de la predicción.

- **Regularización L2**: al igual que la regularización L1, en esta técnica también consiste en añadir un nuevo término en la función de coste, pero en este caso se añade el parámetro “lambda” que multiplica la sume de los cuadrados de los pesos:

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} w_i^2$$

Ilustración 39: Función de coste con Regularización L2 [52].

Siendo también “y” el valor real e “ŷ” el valor de la predicción. La ventaja de esta regularización en comparación con la L1 es que los pesos más grandes en este caso ganan más importancia en comparación con los más pequeños, siendo estos últimos casi nulos. Con ello que se obtiene es un enfoque más preciso en algunas características concretas del modelo, dando menos importancia al resto.

En las arquitecturas de los modelos de extracción de características el bloque II se han utilizado regularizaciones de L2, que resultan muy eficientes para procesamiento de imágenes, remarcando con más fuerza determinadas características de las imágenes.

5. RESULTADOS

Tras la fase de entrenamiento del modelo, se ha procedido a la obtención de los resultados mediante la fase de evaluación y al posterior análisis y conclusiones de estos.

Para la obtención de los resultados de la fase de evaluación, se han realizado distintas pruebas de eficacia de la red y eficiencia o tiempos de inferencia tanto del bloque I como el II. Se ha implementado el código necesario para realizar una evaluación por cada modelo, utilizando el total de las imágenes del conjunto de imágenes de evaluación en cada prueba, todas ellas totalmente distintas a las de la fase de entrenamiento.

Cada una de estas pruebas ha consistido en realizar las inferencias de cada par de bottlenecks de imágenes a partir de las salidas del bloque I. Este par de imágenes ha sido fruto de la combinación equitativa tanto de imágenes que pertenecen a la misma persona como de aquellas que se tratan de personas distintas.

Como se ha definido en el capítulo de desarrollo (arquitectura del Bloque II), se ha establecido un parámetro que permite marcar el grado de similitud que deben tener los rostros de las imágenes a comparar para que puedan considerarse imágenes de la misma persona o de distintas. A este parámetro se le ha llamado "**Umbral de Similitud**". Una evaluación con un Umbral de Similitud elevado indicará que los rostros deben ser más "parecidos" para considerarse como una sola persona al realizar la predicción o inferencia, y viceversa.

En la siguiente ilustración se han representado tres casos con mismos inputs de dos inferencias o comparaciones de imágenes cada uno, obteniendo el mismo valor de similitud al tratarse de las mismas imágenes en todos los casos. La diferencia entre ellos se encuentra en las distintas variaciones del umbral de similitud.

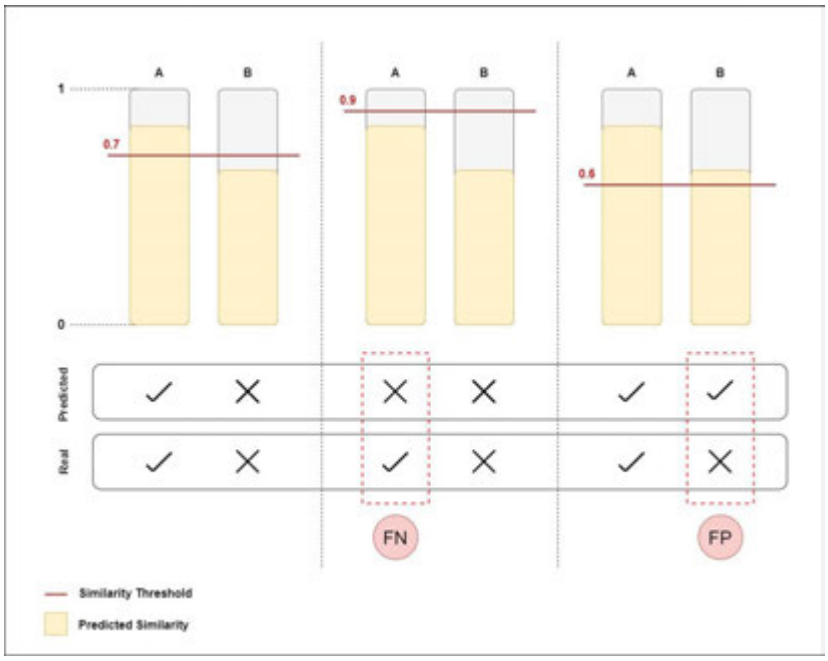


Ilustración 40: Tres evaluaciones de dos inferencias del bloque II cada una, variando el umbral de similitud en cada una de ellas. Los resultados cambian notablemente.

1. En el primer caso, el valor del umbral de similitud se ha establecido de manera que los resultados de predicción coinciden con los resultados reales. La exactitud es del 100%, y la Precisión es del 100%.
2. En el segundo caso, el umbral de similitud ha sido "sobrevalorado". Ambos ejemplos se han considerado como negativos (personas distintas) en los resultados de predicción, mientras que los resultados reales indican que una de las inferencias ha sido realmente imágenes de una misma persona. Se trata de un **Falso Negativo (FN).** La Exactitud ha sido del 50% al haber fallado una inferencia, y la precisión del 0% al no haber considerado ningún verdadero positivo (TP).
3. En el tercer caso, el umbral de similitud ha sido "infravalorado". Ambos ejemplos se han considerado como positivos (mismas personas), cuando realmente una de las comparaciones o inferencias no es realmente la misma persona. Este caso erróneo se considera como un **Falso Positivo (FP),** y es uno de los peores casos que debe evitarse a toda costa, pues en los modelos de verificación biométrica permiten la entrada de intrusos en los dispositivos de verificación por rostros, por ejemplo.

Considerando todos los modelos entrenados en el Bloque II, y combinando los distintos hiperparámetros utilizados, se obtienen un total de 576 modelos distintos, anteriormente calculados en la fase de entrenamiento.

Con estos modelos, se han ido obteniendo las predicciones de cada uno de estos modelos a través del dataset de evaluación. Además, se han realizado diversos tipos de entrenamientos por cada modelo, donde se ha mantenido constante cualquier parámetro, a excepción del umbral de similitud, variando entre distintos valores con el fin de lograr los resultados necesarios para encontrar el valor óptimo que logre maximizar la exactitud y al mismo tiempo minimizar la tasa de falsos positivos.

Por tanto, se han obtenido resultados por cada uno de los modelos dependiendo del valor del umbral de similitud, que varía entre 0.75, 0.7, 0.75, 0.8, 0.9, 0.95, 0.97, 0.99, 0.995 y 0.999. El número de evaluaciones totales ha sido de:

- $\boldsymbol{Total\ Evaluaciones\ Bloque\ II} = Nº\ de\ Modelos * Nº\ Valores\ de\ Umbral\ de\ Similitud = 576 * 10 = 5760\ evaluaciones$

La obtención del bottleneck de cada imagen del Bloque I, al igual que en la fase de entrenamiento, se ha obtenido únicamente una sola vez por cada una de las 4 arquitecturas. Por tanto, el número total de inferencias realizadas del Bloque I ha sido de:

- $\boldsymbol{Total\ inferencias\ de\ evaluación\ Bloque\ I} = Nº\ de\ Arquitecturas\ Bloque\ I * Nº\ total\ de\ imágenes\ de\ evaluación = = 4 * 9638 = \mathbf{38552}\ inferencias\ bloque\ II$

Aunque parezca un número bastante elevado, gracias a la separación por bloques se ha podido optimizar el número total de inferencias con alto coste computacional, en concreto en las 4 arquitecturas de redes neuronales convolucionales del bloque II, más profundas y complejas que la red de neuronas del bloque II. De no haber realizado esta técnica, el número total de inferencias del bloque I que se tendrían que haber realizado serían:

- $\boldsymbol{Total\ inferencias\ de\ evaluación\ Bloque\ I\ (caso\ ineficiente)} =$
  $Nº\ Modelos\ Totales * Nº\ Umbrales\ de\ similitud\ por\ evaluación\ *$
  $Nº\ túplas\ evaluación\ Bloque\ II\ por\ modelo *$
  $Nº\ Imágenes\ por\ túpla\ Bloque\ II =$
  $= 576 * 10 * 11306 * 2 = \mathbf{130245120}\ inferencias\ totales\ de\ Bloque\ I$

Comparando ambos resultados, realizando esta técnica de separación por bloques se ha logrado que el modelo sea 130245120/38552=3378.42706 veces más eficiente, realizando una sola inferencia por cada imagen y modelo para cualquier evaluación posterior.

5.1. PLANTEAMIENTO MÉTRICAS UTILIZADAS PARA LOS RESULTADOS

De los modelos entrenados y evaluados, se han obtenido las matrices de confusión y los tiempos de ejecución de cada uno de ellos. Los resultados obtenidos directamente tras las evaluaciones realizadas han sido los siguientes:

1. Eficacia:
   - **Verdaderos Positivos (VP)** o True Positives (TP): es el número total en la fase de evaluación de ejecuciones de la red donde el valor de predicción es positivo (las imágenes pertenecen a la misma persona) y el valor real también lo es.
   - **Verdaderos Negativos (VN)** o True Negatives (TN): es el número total en la fase de evaluación de ejecuciones de la red donde el valor de predicción es negativo (las imágenes no pertenecen a la misma persona) y coincide con el valor real, negativo también.
   - **Falsos Negativos** o False Negatives **(FN) (Error Tipo II):** es el número total de ejecuciones de la red en la fase de evaluación donde el valor de predicción es negativo, pero el valor real es positivo. Esto es el caso donde dos imágenes de una misma persona no se han considerado en la predicción como las mismas.
   - **Falsos Positivos** o False Positives **(FP) (Error Tipo I):** es el número total de ejecuciones de la red en la fase de evaluación donde el valor de predicción es positivo, pero realmente el valor real es negativo. Esto es el caso donde dos imágenes de personas distintas se han considerado como las mismas en la predicción.
2. Eficiencia:
   - **Tiempo de Entrenamiento** o Training Time (TT): es el tiempo total que requiere la fase de entrenamiento del bloque II por cada modelo.
   - **Tiempo de Evaluación** o *Evaluation Time* (ET): es el tiempo total que requiere realizar la evaluación de cada uno de los modelos, por cada uno de los umbrales de similitud establecidos. Se debe tener en cuenta que esta evaluación es

únicamente a través de inferencias del bloque II. El tiempo de ejecución o de inferencia final por cada input en el modelo general deberá utilizar el tiempo de pre-procesamiento, el tiempo de inferencia del bloque I y el tiempo del bloque II.

Con estos resultados directos, se han podido establecer una serie de resultados basados en nuevas métricas, capaces de interpretar los modelos de una manera más precisa y razonada. Se incluyen a continuación la descripción junto con la ecuación utilizada en cada una de ellas:

- **Exactitud** o **Accuracy** (**ACC**): indica la proporción del total de aciertos por parte de las predicciones, tanto para casos positivos (misma persona) como para casos negativos (personas distintas).

$$ACC = \frac{VP + VN}{VP + FP + FN + VN} = \frac{Total\ Predicciones\ Correctas}{Total\ Predicciones}$$

- **Precisión** o Valor Predictivo Positivo: determina la proporción de los valores acertados donde la predicción se ha considerado como positiva. Una mayor precisión indicará una menor proporción de falsos positivos.

$$Precisión = \frac{VP}{VP + FP} = \frac{Positivos\ Acertados}{Total\ Positivos\ de\ Predicciones}$$

- **Sensibilidad** o **Recall**, o **Tasa de Verdaderos Positivos**: indica la proporción de valores reales positivos donde el modelo ha acertado en sus predicciones.

$$Recall = \frac{VP}{VP + FN} = \frac{Positivos\ Acertados}{Total\ Positivos\ Reales}$$

- **Especifidad o Tasa de Verdaderos Negativos:** se trata de los casos negativos reales que el algoritmo ha clasificado correctamente.

$$Especifidad = \frac{VN}{VN + FP} = \frac{Negativos\ Acertados}{Total\ Negativos\ Reales}$$

- **Tasa de Falsos Positivos:** indica la proporción de falsos positivos sobre el total de valores negativos reales.

$$TFP = \frac{FP}{VN + FP} = \frac{Falsos\ Positivos}{Total\ Negativos\ Reales} = 1 - Esfecifidad$$

- **Tasa de Falsos Negativos:** indica la proporción de falsos negativos sobre el total de positivos reales.

$$TFN = \frac{FN}{VP - FN} = \frac{Falsos\ Negativos}{Total\ Positivos\ Reales} = 1 - Recall$$

- **F1Score:** combina la precisión y la sensibilidad de manera harmónica en una sola métrica, donde el modelo generalmente será más preciso y eficaz cuando mayor es su valor.

$$F_1Score = 2 * \frac{Recall * Precisión}{Recall + Precisión}$$

El F1Score es un valor a tener en cuenta en clasificación binaria, ya que permite encontrar un balance entre la Precisión y el *Recall*, siempre y cuando la Precisión y el *Recall* tengan el mismo peso u importancia. En este caso, la tasa de Falsos Positivos (y por tanto la Precisión) implica una mayor importancia que la tasa de Falsos Negativos (y por tanto el *Recall*), ya que, como se ha indicado anteriormente, en modelos de verificación biométrica, un aumento en la seguridad de los dispositivos mediante la denegación de accesos a personas ajenas se considera más crítico en comparación con la denegación de accesos por error por personas realmente permitidas en un sistema. En otras palabras, la importancia de un caso donde se deniega el acceso por error a una persona autorizada en un dispositivo (FN) no es la misma que otro caso donde se concede el acceso a una persona no autorizada (FP).

Por ello, existe una variación de la F1Score, la ***FBetaScore***, donde es posible dar más importancia a cada una de las métricas. Chinchor [53] define la ecuación que será posible aplicarla en los resultados obtenidos:

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \qquad (0 \leq \beta \leq +\infty).$$

Ecuación 1: Ratio F_β [53].

Donde "P" es la precisión, "R" es el *Recall*, y "β" es el valor que controla el equilibrio de importancia entre P y R. Una B igual a uno indica que la P y R tienen el mismo grado de importancia a la hora de comprobar los resultados obtenidos (F1Score convencional); si B > 1, el valor final estará más influenciado por el *Recall*; y si B < 1, el valor final estará más influenciado por la Precisión.

El objetivo de este estudio es, por tanto, además de mejorar el tiempo de ejecución, encontrar un modelo que logre minimizar la tasa de Falsos Positivos y Falsos Negativos, dando más importancia a la reducción de la tasa de Falsos Positivos. Por tanto, basado en los artículos más destacados [54], se ha establecido una beta de "0,5", dando más peso a la Precisión.

La ecuación de $\mathbf{F_{0,5}}$ por tanto queda:

$$F_{0,5} = \frac{((0{,}5)^2 + 1)PR}{(0{,}5)^2\mathrm{P} + \mathrm{R}} = \frac{1{,}25PR}{0{,}25\mathrm{P} + \mathrm{R}} = 5\frac{PR}{\mathrm{P} + 4\mathrm{R}}$$

Además de la $F_{0,5}$, numerosos estudios además han utilizado la curva **ROC** o Característica Operativa del Receptor ("***Receiver Operating Characteristic***") [45] [50] [43] [44] [34]. Esta curva representa gráficamente la sensibilidad de los resultados frente a la especifidad en los sistemas de clasificación binarios, en función del umbral de sensibilidad o, en otras palabras, establece la relación entre la tasa de verdaderos positivos (VPR) frente a la tasa de falsos positivos (FPR) en función del umbral de discriminación que se establezca. Esta curva sirve de gran ayuda para establecer cuáles de los modelos a analizar son los más óptimos, siempre y cuando además el

número de muestras de cada clase sea equivalente. En este caso, el número de clases es completamente el mismo, 50% misma persona, 50% personas distintas.

Con la curva ROC definida, se ha podido utilizar otra métrica que facilita el análisis de los distintos modelos permitiendo obtener una información general cada una de las curvas ROC obtenidas: el área sobre la curva ROC o **AUC** ("*Area Under the Curve*") [55]. Esta área puede resultar de ayuda, no obstante, no se trata de una medida definitiva para establecer conclusiones finales de los modelos, ya que podría darse el caso de que en un umbral de discriminación concreto los resultados son atípicamente positivos, mientras que en el resto de valores de este umbral no son del todo aceptables, y el AUC por tanto no es tan elevado como es de esperar (no se tienen en cuenta los valores atípicos que puedan resultar realmente beneficiosos).

5.2. RESULTADOS DE LAS MÉTRICAS

En base a los resultados obtenidos de todos los modelos entrenados y evaluados, se ha obtenido una tabla dinámica en Excel que permite agrupar los resultados de múltiples maneras. Los resultados se han podido agrupar en dos grupos: eficacia, donde se han obtenido los resultados tales como la $F_{0.5}$, las curvas ROC o el área bajo la curva AUC; y eficiencia, donde los valores indican los tiempos y velocidades de cada una de las arquitecturas.

5.2.1. EFICACIA

Este grupo está formado por aquellas métricas que permiten definir la calidad de las predicciones en función de correctos aciertos y la reducción de los fallos de predicción.

En primer lugar, tras la observación de los resultados obtenidos, se ha realizado un análisis sobre los distintos parámetros de entrada explicados anteriormente (tasa de aprendizaje, "*batch_size*", "*max_steps*", etc.). De todos ellos, la tasa de aprendizaje ha puesto en situación muy diferente los resultados obtenidos dependiendo de qué valor ha sido utilizado, creando una distorsión de los resultados muy remarcada. Por tanto, se ha tratado de realizar un primer análisis de los resultados sobre este primer hiperparámetro. Tras la obtención de varios resultados en un primer momento, se ha decidido utilizar los valores de "0.001" y "0.0001" al ser los más óptimos. Los resultados obtenidos a partir de estos dos valores han sido los siguientes:

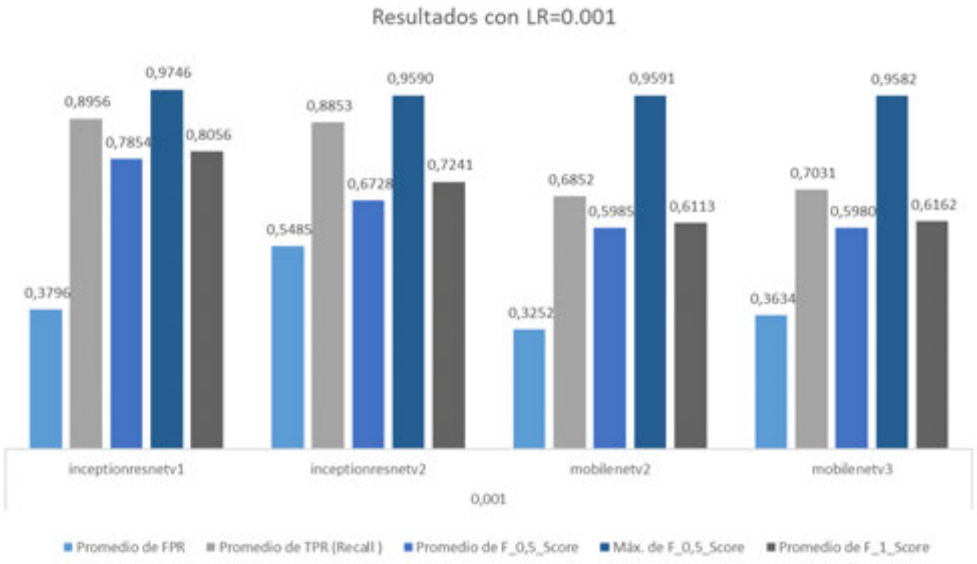


Ilustración 41: comparativa resultados con Learning Rate igual a 0,001.

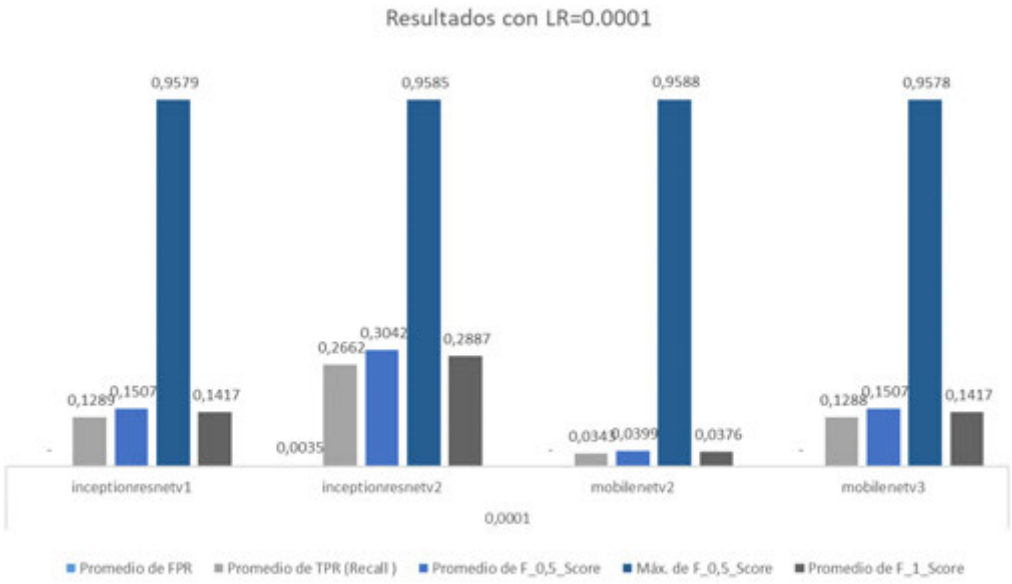


Ilustración 42: comparativa resultados con Learning Rate igual a 0,0001.

Ambas gráficas muestran las medias de las métricas de FPR, TPR la $F_{0.5}$ y la F_1 de todas las simulaciones de cada uno de los modelos en función de la tasa de aprendizaje aplicada, además del valor máximo de $F_{0.5}$ obtenido por una de las simulaciones de cada una de las distintas arquitecturas. Se puede ver claramente cómo la tasa de aprendizaje de 0.001 utilizada en el Bloque II del modelo supera a cualquiera de los resultados con la tasa de aprendizaje de 0.0001. Aunque en la segunda gráfica el promedio de FPR es mayor, y por tanto negativo para el análisis,

realmente los valores de F_1 y $F_{0.5}$ son ambos superiores a la primera gráfica, tanto los valores promedio como el valor máximo de $F_{0.5}$.

5.2.1.1. $F_{0.5}$

A continuación, se han representado las distintas métricas de la $F_{0.5}$ a partir de los resultados obtenidos en las simulaciones en función de cada una de las arquitecturas a través de sus *BoxPlot*:

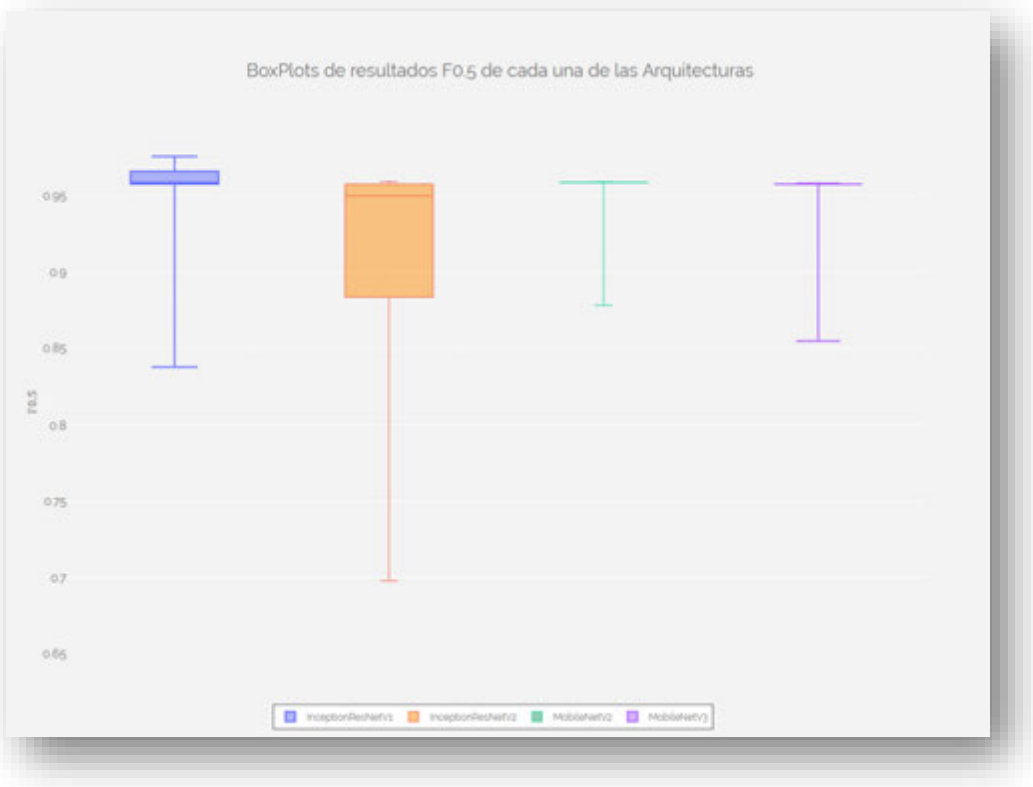


Ilustración 43: BoxPlots de resultados $F_{0,5}$ de cada una de las Arquitecturas

En cuanto al valor medio de $F_{0.5}$, se han obtenido varias conclusiones:

- Las redes de Inception-ResNet han obtenido mejores resultados que las redes de MobileNet.
- Todas las redes tienen una varianza de esta métrica relativamente similar, posiblemente al tratarse de simulaciones en cada una de las cuatro arquitecturas con mismos conjuntos de inputs e hiperparámetros, mostrando un comportamiento similar en cuanto a la red neuronal utilizada del bloque II.
- Sin embargo, la métrica en esta gráfica que realmente ofrece clara información ha sido el valor de $F_{0.5}$ máximo por arquitectura del Bloque II. En cada una de estas arquitecturas, el valor máximo ha sido de 0.9746 para la red de Inception-ResNetV1, y 0.9590 aproximadamente para el resto de redes, valores muy similares para todas ellas.

Este valor resulta bastante destacable al poder establecer los hiperparámetros e inputs de una sola simulación que permitan este valor máximo, alcanzando niveles de eficacia de la clasificación binaria muy positivos.

En la tabla siguiente se muestran los valores de entrada establecidos para obtener unos resultados tales que maximicen la $F_{0.5}$ por cada una de las cuatro arquitecturas:

Arquitectura	Batch_Size	Max_Steps	Dropout_Keep_Prob	Learning_Rate	Discrimination_Threshold	Semilla	Máx. de F_0,5_Score
Inception-ResNet-V1	8	500	0.85	0.001	0.7	29	0,974555476
Inception-ResNet-V2	16	250	0.85	0.001	0.97	13	0,959028207
MobileNet-V2	8	250	0.85	0.001	0.7	13	0,959071388
MobileNet-V3	16	250	0.85	0.001	0.65	80	0,958183843

Ilustración 44: parámetros de cada arquitectura que maximizan la $F_{0.5}$.

5.2.1.2. CURVA ROC

Con los valores máximos obtenidos anteriormente, se ha procedido a estudiar estas simulaciones concretas por cada arquitectura que haya logrado estos resultados. Para ello, se han trazado las curvas ROC de cada una de ellas a continuación, a partir de los distintos valores de TPR y FPR en función del umbral de similitud establecido:

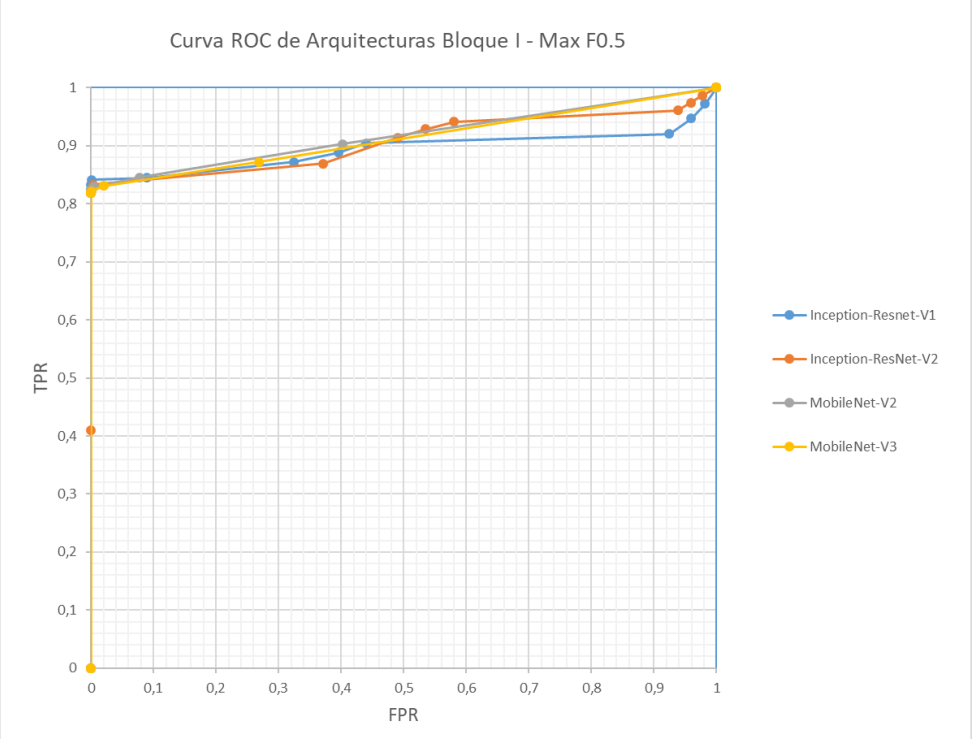


Ilustración 45: Curva ROC obtenida con los parámetros que maximizan la $F_{0.5}$ en cada arquitectura.

En ella se encuentran los valores obtenidos de la tasa de verdaderos positivos en función de la tasa de falsos positivos, para las simulaciones de los modelos que han maximizado la $F_{0.5}$.

Los resultados de la gráfica de los cuatro modelos muestran con un mismo comportamiento y parecido, una cierta tendencia desequilibrada entre la TPR y la FPR. Demuestran cómo la FPR trata de ser minimizada en cada uno de los valores del umbral de similitud (con un valor de TRP de 0 hasta poco más de 0.8, la FPR es nula). Con ello lo que se permite es reducir al máximo los falsos positivos, y al mismo tiempo aumentar todo lo posible los verdaderos positivos frente a los falsos negativos, dando más prioridad o importancia a lo primero.

En cuanto a los modelos ajenos al trabajo, los modelos propuestos en este documento superan ligeramente este tipo de técnicas. Cabe destacar que la tasa de falsos positivos, al haber priorizado la minimización de la misma, ha logrado valores prácticamente nulos manteniendo valores de la tasa de verdaderos positivos hasta los 0.85 puntos (85%), algo que los modelos ajenos tradicionales no consiguen lograr.

5.2.1.3. ÁREA BAJO LA CURVA AUC

Se han obtenido los valores del área bajo la curva AUC de cada uno de los modelos anteriores en la siguiente ilustración:

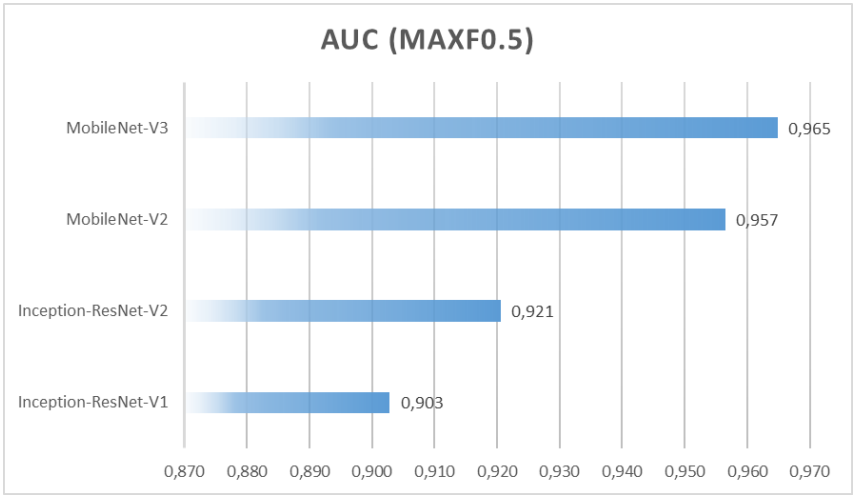


Ilustración 46: Áreas bajo la curva ROC de cada arquitectura mediante los parámetros que maximizan la $F_{0.5}$.

A pesar de haber considerado como valores muy similares los considerados en la gráfica de las curvas ROC, el área bajo la curva de cada una de ellas descubre pequeñas diferencias en cada una de ellas, destacando la arquitectura de MobileNet-V3 con un AUC de mayor valor, seguida de su hermana MobileNetV2, y por último Inception-ResNet-V2 y V1. El AUC indica la eficacia de las curvas ROC de manera general, incluyendo todos los umbrales de similitud utilizados. Por esto mismo, a pesar de que la red de Inception-ResNet-V1 tiene la AUC de menor valor, existen resultados puntuales con un umbral de similitud concreto que permite superar el resto de arquitecturas. Se ha podido comprobar en la tabla anterior cómo a través de la semilla 29 y utilizando un umbral de similitud de 0.7 y los hiperparámetros de la tabla $F_{0.5}$ ha sido de 0.975 aproximadamente, superando al resto de arquitecturas.

5.2.1.4. EXACTITUD Y PRECISIÓN DE LOS MODELOS MÁS DESTACADOS.

Utilizando los modelos que se han utilizado en los anteriores gráficos, tablas y diagramas, se han obtenido los resultados de las métricas de Exactitud y Precisión de cada una de ellas.

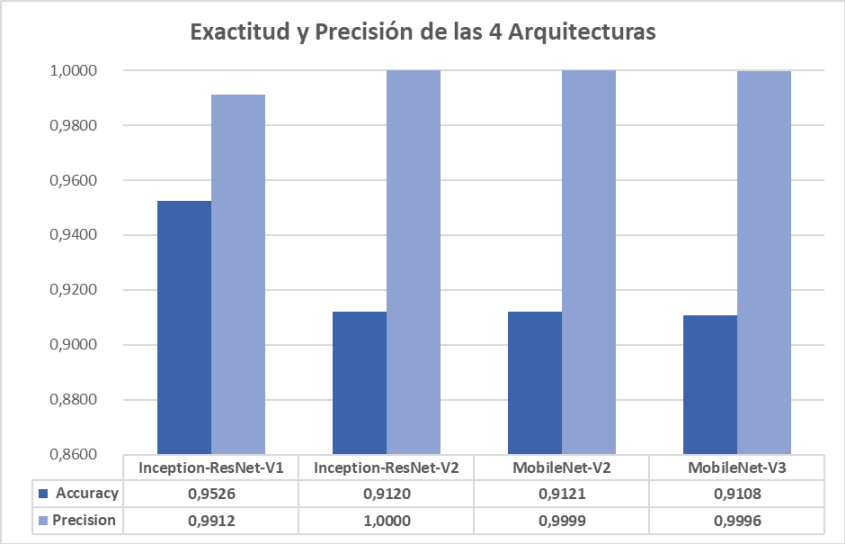


Ilustración 47: Exactitud y precisión de cada arquitecturas mediante los parámetros que m maximizan la $F_{0.5}$.

Este último diagrama sobre la eficacia de cada red puede demostrar cómo prácticamente los resultados son muy similares entre la red de Inception-ResNet-V2 y las dos redes de MobileNet-V2 y V3. Por otro lado, la red de Inception-ResNet-V1 muestra una diferencia con respecto al resto de arquitecturas, disminuyendo ligeramente su precisión a cambio de un gran aumento de la exactitud (*Accuracy*). Comprobando la $F_{0.5}$, su valor es ligeramente superior al resto, por lo que el resultado resulta muy positivo si solamente es considerado esta simulación en concreto para esta arquitectura. En otras palabras, la configuración de parámetros para Inception-ResNet-V1 permite reducir una proporción de falsos negativos bastante elevada con respecto al resto de arquitecturas, con la parte negativa donde la proporción de falsos positivos se ve ligeramente aumentada.

5.2.2. EFICIENCIA

Una vez analizada la eficacia de cada arquitectura, es importante realizar un análisis de rendimiento de cada uno de ellos. Para ello, se han realizado evaluaciones de tiempos en cada uno de los dos bloques del modelo. En primer lugar, se han comprobado el tiempo que ha requerido la obtención de los todos cuellos de botella de cada una de las cuatro arquitecturas, a partir del bloque I del modelo. A continuación, se ha procedido a calcular los tiempos de entrenamiento y evaluación de cada una de las arquitecturas realizadas en el bloque II. Los resultados son los siguientes:

Tabla Eficiencia	Bloque I	Bloque II			
Arquitectura	Tiempo Inferencia (s)	Promedio Train_Time (s)	Desvest Train_Time (s)	Promedio Eval_Time (s)	Desvest Eval_Time (s)
Inceptionresnetv1	755,5570	47,6871	17,5195	39,1580	2,9846
Inceptionresnetv2	1.688,8550	60,4338	15,7595	14,4920	1,8415
Mobilenetv2	**327,4200**	**43,2810**	15,6505	38,9388	5,2447
Mobilenetv3	418,3870	44,2898	15,8942	**13,2108**	1,2547

Tabla 6: tiempos de ejecución de cada una de las arquitecturas.

La tabla anterior muestra, por un lado, el tiempo en segundos que ha sido necesario para poder realizar todas las inferencias de las imágenes de ambos datasets (entrenamiento y evaluación) en cada una de las distintas arquitecturas. Se puede ver cómo las arquitecturas de Inception-ResNet, con un tamaño superior y con mayor complejidad, han realizado todas las inferencias en un tiempo bastante superior que las arquitecturas de MobileNet, más ligeras y con un tamaño inferior. Por otro lado, la tabla además muestra el tiempo total medio que ha requerido realizar los entrenamientos del clasificador binario del bloque II, además de las posteriores evaluaciones en función del umbral de similitud. El bloque II, al tener como input un bottleneck muy parecido en todas las arquitecturas, y al poseer la misma implementación de la red neuronal, lo único que ha podido influir en cada uno de ellos ha sido la posible pérdida o retraso en función de los pesos que se han ido guardando, que variarán según la arquitectura. No obstante, existe una diferencia de tiempos claramente separadora entre Inception-ResNet-V1 y MobileNet-V2 (versiones anteriores) y entre Inception-ResNet-V2 y MobileNet-V3 (versiones posteriores), con tiempos de evaluación de 39 y 14 segundos aproximada y respectivamente.

A continuación, se ha realizado el cálculo necesario para obtener los tiempos totales aproximados que requieren cada una de las arquitecturas para poder realizar una predicción completa: desde la entrada al modelo de imágenes listas para ser procesadas, hasta la obtención de un resultado que indique si el par de imágenes pertenece a la misma persona o no. Para ello, se ha utilizado la tabla anterior, junto con algunos datos conocidos para poder realizar este cálculo:

Tabla Eficiencia Por Inferencia Completa	Bloque I	Bloque II		Total
Arquitectura	Tiempo Inferencias Img1 + Img2 (s)	Promedio Eval_Time (s)	Desvest Eval_Time (s)	Total Tiempo
Inceptionresnetv1	0,1212	0,0035	0,0003	**0,1247**
Inceptionresnetv2	0,2710	**0,0013**	0,0002	**0,2722**
Mobilenetv2	**0,0525**	0,0034	0,0005	**0,0560**
Mobilenetv3	**0,0671**	**0,0012**	0,0001	**0,0683**

Tabla 7: : tiempos de ejecución por cada inferencia de cada bloque de cada una de las arquitecturas.

Se pueden ver los tiempos requeridos por cada par de imágenes en cada uno de los bloques, junto con el tiempo total, resultado de la suma de ambos valores. Para cada predicción, el tiempo medio requerido de las inferencias del bloque I para cada arquitectura viene dado por:

$$Tiempo\ Inferencias\ Bloque\ I = 2 * \frac{Tiempo\ total\ inferencias\ Bloque\ I}{Nº\ Imgs\ Dataset\ Entrenamiento + Nº\ Imgs\ Dataset\ Evaluación}$$

Y para el bloque II, el tiempo promedio de cada predicción en el bloque II viene dado por:

$$Tiempo\ Promedio\ Inferencia\ Bloque\ II = \frac{Tiempo\ promedio\ por\ Evaluación\ del\ Bloque\ II}{Nº\ inferencias\ bloque\ II\ por\ cada\ evaluación}$$

El mismo concepto se ha utilizado para el cálculo de la desviación estándar promedia de los tiempos de evaluación únicamente del bloque II:

$$Desv\ Estándar\ Inferencia\ Bloque\ II = \frac{Desv\ Estándar\ del\ Tiempo\ por\ Evaluación\ del\ Bloque\ II}{Nº\ inferencias\ bloque\ II\ por\ cada\ evaluación}$$

Cabe mencionar que el cálculo del tiempo por cada inferencia del bloque I en la predicción se incluye una multiplicación por dos debido a la necesaria obtención del bottleneck para el bloque II dos veces, una por cada imagen de entrada. Sin embargo, este valor está considerado como el peor de los casos, considerando el hardware utilizado como un sistema de procesamiento secuencial en lugar de paralelo, donde se procesa una imagen a continuación de otra. De ser un sistema paralelo este valor se reduciría considerablemente.

Los valores de tiempos de predicción totales se muestran por tanto en la última columna de la tabla anterior. Se concluye este análisis de rendimiento con las redes de MobileNet como las más “rápidas”, que permiten realizar predicciones y obtener resultados en un tiempo realmente menor que las redes de Inception-ResNet. Los tiempos de estas últimas (0,12 y 0.27 segundos) son alrededor de dos e incluso tres veces superiores a los de las redes de MobileNet (0,056 y 0,068 segundos).

6. PLANIFICACÍON Y PRESUPUESTO

En este capítulo se ha elaborado la planificación y presupuesto de un posible proyecto que permita la puesta en funcionamiento de una actividad económica futura. La idea estimada del supuesto proyecto estaría apoyada un sistema similar al elaborado en el anexo *Proceso de Aplicación Biométrica*.

6.1. PLANIFICACIÓN

La planificación del proyecto se ha ido marcando principalmente a través de pequeños objetivos semanales, a su vez agrupados por pequeñas con un objetivo común diferenciado. Los objetivos semanales se han ido estableciendo en función del progreso realizado por el alumno en función de lo que se haya avanzado en previas semanas. Las épicas no han tenido una duración estrictamente fijada, sino que ha dependido del tamaño del objetivo a lograr. Las fases o etapas principales que ha seguido este trabajo han sido las siguientes:

1. **Análisis de redes neuronales**. Se ha elaborado un estudio exhaustivo del estado del arte sobre la inteligencia artificial, redes de neuronas convolucionales, etc.
2. **Profundización en lenguaje de programación utilizado**. En este caso, el alumno se ha formado previamente a la elaboración del trabajo en el lenguaje de Python, junto con librerías como Tensorflow, Keras, OpenCV, etc.
3. **Primeros acercamientos sobre biometría facial**. Se han elaborado pequeñas implementaciones y pruebas en un primer momento sobre técnicas de clasificación de imágenes a través del aprendizaje automático. Se han utilizado clasificadores como "*Support Vector Machines*" (SVM), pequeños ejemplos de clasificación de imágenes como la clasificación de dígitos numéricos (MNIST), etc.
4. **Análisis arquitecturas de redes de neuronas convolucionales (CNN) más utilizadas para reconocimiento de imágenes.** En esta fase se ha comenzado a utilizar arquitecturas de redes entrenadas para posteriormente realizar inferencias de clasificación de imágenes, en concreto la clasificación facial. Entre ellas destacan las redes de Inception, ResNet, MobileNet, Inception-ResNet, etc.
5. **Diseño de red siamesa.** Una vez obtenidos los conocimientos previos necesarios, se ha procedido a realizar las primeras versiones de la arquitectura global de los modelos utilizados.
6. **Implementación red siamesa.** Tras el diseño de la red final utilizada se ha procedido a la implementación del código necesario.
7. **Análisis, diseño e implementación de pruebas:** en esta fase se ha realizado en primer lugar un pequeño estudio e investigación sobre las técnicas de evaluación más utilizadas en cuanto a la verificación biométrica facial. Más tarde se ha procedido a la definición de las métricas más apropiadas para este estudio, y su posterior implementación y ejecución.
8. **Automatización de pruebas:** debido al gran tamaño del número de pruebas realizadas ente todos los modelos y sus posibles parámetros utilizados, se ha decidido dedicar una

fase a la automatización de las mismas, elaborando un pequeño diseño y su posterior implementación.

9. **Documentación y realización de la memoria.** Se ha procedido en esta fase al desarrollo de la memoria del trabajo de fin de grado, a partir de todo lo aprendido y elaborado en el proyecto.

En el caso de realizar este proyecto a nivel empresarial para la implementación en un dispositivo móvil, se va a proceder a realizar una planificación de las tareas de manera que se puedan realizar en un tiempo apropiado.

6.1.1. METODOLOGÍA

La metodología de trabajo más adecuada para este proyecto ha sido una de las más utilizadas en el momento, se trata de la metodología ágil. Gracias a ella, muchas empresas aprovechan el gran avance de múltiples herramientas que permiten un mayor flujo de la comunicación y control entre los equipos. Además, con la metodología ágil, las empresas obtienen múltiples ventajas: se logra minimizar el peso de cada una de las tareas consiguiendo así aumentar la eficiencia de los trabajadores y, por tanto, el coste total del proyecto; también se logra una mayor flexibilidad en cuanto a la realización de continuos cambios que puedan darse en cualquier momento, gracias a que los trabajadores son capaces de estar informados en todo momento ante cualquier decisión; además, la interacción con el cliente es continua, permitiendo así conocer en todo momento las necesidades de cada cliente y reaccionar ante ellas.

Basándose en las etapas o fase anteriormente descritas, y considerando el conocimiento adquirido por este trabajo, se han diseñado cada una de las iteraciones necesarias para el desarrollo del proyecto. Cada una de ellas estará compuesta por varias tareas principales: ingeniería de requisitos, diseño e implementación, y realización de pruebas.

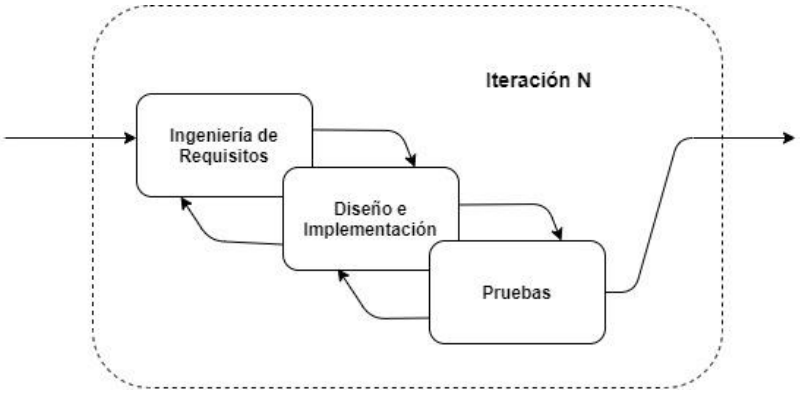


Ilustración 48: Metodología agile utilizada.

Es importante comentar que la fase de ingeniería de requisitos de una iteración se podrá empezar con anterioridad a la finalización del diseño y la implementación y las pruebas de la

iteración anterior, con el fin de agilizar los procesos entre iteraciones y evitar posibles bloqueos. En el diagrama de Gantt posterior se podrá observar más detenidamente esta técnica

6.1.2. PLANIFICACIÓN

Se han estimado los siguientes roles para el desarrollo del proyecto:

- **Ingeniero de software**: tomará además las tareas de jefe de proyecto, además de ayudar en la ingeniería de requisitos y el diseño
- **2 desarrolladores/programadores**: serán los encargados del diseño, implementación y pruebas de cada iteración. Serán apoyados a su vez por el ingeniero de software.

Todos los empleados trabajarán una jornada completa de 40 horas semanales, basándose en la metodología ágil por iteraciones. Cada iteración tendrá una duración de una o dos semanas aproximadamente. Estará formada de la siguiente manera:

Iteración 1: Toma de contacto e iniciación.

- Ingeniería de Requisitos: se realizará un primer análisis de contexto de las tecnologías más utilizadas de biometría facial. A su vez, se elaborará un pequeño plan de requisitos mínimo para el uso de estas tecnologías para entrar en situación y poder familiarizarse con las herramientas utilizadas y con el equipo.
- Diseño e implementación: se realizará la primera toma de contacto mediante el desarrollo de un pequeño ejemplo de clasificación facial en el lenguaje utilizado de Python. Con ello los desarrolladores podrán agilizar sus habilidades para posteriores iteraciones.
- Pruebas: se realizarán pequeñas pruebas en base a este primer desarrollo base que puedan ajustar las técnicas que se utilizarán posteriormente en las siguientes iteraciones.

Iteración 2: Mínimo Producto Viable. Primer modelo básico.

- Ingeniería de Requisitos: se realizará un análisis de los requisitos necesarios y la planificación para poder diseñar e implementar un primer modelo de verificación facial
- Diseño e implementación: se procederá al diseño de una arquitectura básica que permita realizar inferencias de verificación facial entre imágenes de personas, y su posterior implementación.
- Pruebas: se verificará el correcto funcionamiento de este modelo.

Iteración 3: Modelo avanzado. Red siamesa.

- Ingeniería de Requisitos: se elaborará un plan de requisitos parecido al de la anterior iteración.
- Diseño e implementación: se elaborará el diseño y la implementación de la arquitectura más compleja formada por la red neuronal convolucional siamesa.

- Pruebas: se realizarán las pruebas necesarias para comprobar que el modelo funcione correctamente y permita cumplir su función principal.

Iteración 4: Automatización de pruebas y monitorización.

- Ingeniería de Requisitos: se establecerán los requisitos necesarios para elaborar un sistema que permita realizar el plan de pruebas de manera automatizada, pudiendo realizar entrenamientos de modelos y evaluaciones sin tener que depender día y noche de personas que se encarguen de ello. Se realizará además un pequeño sistema de monitorización que permita conocer en todo momento el estado de los modelos y sus evaluaciones.
- Diseño e implementación: se procederá a el diseño de esta automatización y monitorización.
- Pruebas: se realizarán pruebas de la automatización antes de ponerse en funcionamiento en el proyecto, además de comprobar la correcta monitorización.

Iteración 5: Entrenamiento completo del modelo definitivo.

- Ingeniería de Requisitos: se establecen las especificaciones para que el modelo definitivo utilizado a través de las pruebas previas realizadas pueda ser entrenado con la totalidad de los datos disponibles.
- Diseño e implementación: se elaboran los diseños necesarios para poder entrenar el modelo a través de todos estos nuevos datos de entrenamiento, así como su posterior implementación.
- Pruebas: se realizarán todas la pruebas y evaluaciones necesarias una vez el modelo haya sido entrenado, y la comprobación de los buenos resultados esperados.

Iteración 6: Despliegue en los sistemas finales.

- Ingeniería de Requisitos: es esta fase se establecen las especificaciones para elaborar un despliegue en dispositivos móviles, una vez comprobados los resultados favorables para la implantación en los mismos.
- Diseño e implementación: se procederá a diseñar e implementar la estructura y funcionamiento del modelo en estos dispositivos: herramientas utilizadas, diagramas de flujo en el sistema móvil y su desarrollo final.
- Pruebas: se verificará el funcionamiento a través de pruebas con los dispositivos móviles, desde los más flexibles de incorporar hasta los que disponen de las mayores limitaciones y restricciones.

A continuación, se expone el diagrama de Gantt previsto para la planificación del proyecto, asumiendo una fecha de inicio de 31 de agosto. La duración de las tareas es directamente proporcional al peso de cada una de ellas. La fecha fin estimada del proyecto es el 19 de noviembre de 2020, ocupando un total de 2 meses y 20 días aproximadamente.

Ilustración 49: Diagrama de Gantt del proyecto.

6.2. PRESUPUESTO

Para la realización del presupuesto del proyecto se han estimado los costes de los empleados y los costes de material, ya sean licencias de hardware y licencias y servicios software.

6.2.1. COSTES DE PERSONAL

En base al apartado de la planificación anterior, el personal está formado por un ingeniero de software y dos desarrolladores o programadores a tiempo completo. El ingeniero y uno de los desarrolladores tendrán un nivel experimentado avanzado, y el desarrollador restante tendrá un nivel medio-bajo.

Los costes de los empleados estimamos son:

Costes Empleados	Coste/hora	Horas semanales	Total semanas	Total horas	Total Coste
Ingeniero de Software - Nivel Avanzado	30 €	40	12	480	14.400 €
Desarrolador - Nivel Avanzado	25 €	40	12	480	12.000 €
Desarrolador - Nivel Medio/Bajo	12 €	40	12	480	5.760 €
Total					**32.160 €**

Tabla 8: Costes del proyecto de los empleados.

6.2.2. COSTES DE MATERIAL

Además de los empleados, se han estimado los costes de todo el material utilizado. Se han necesitado de los ordenadores portátiles, los teléfonos móviles para la implementación del modelo en ellos el último mes del proyecto, las licencias de software para el desarrollo del proyecto, el alquiler del local y los gastos de luz.

Costes materiales	Precio Unitario	Unidades	Meses	Amortizacion/mes	Total Coste
Local + Luz	300,00 €	1	3		900,00 €
Ordenador Portátil Lenovo ThinkPad T580	1.107,41 €	3	3	39,55 €	3.440,88 €
Telefono Movil	300,00 €	2	1	7,14 €	607,14 €
Google Cloud Platform API	20,00 €	1	3		60,00 €
Licencia Software PyCharm	16,67 €	3	3		150,00 €
Licencia Software GitHub	4,00 €	3	3		36,00 €
Licencia Software Jira	14,00 €	3	3		126,00 €
Total					**4.420,02 €**

Tabla 9: Costes del proyecto de material.

6.2.3. RESUMEN DE COSTES

Resumen Costes	Coste
Costes Empleados	32.160,00 €
Costes materiales	4.420,02 €
Total Costes Proyecto	**36.580,02 €**

Tabla 10: Resumen costes totales de proyecto.

El coste total de llevar a cabo el proyecto será por tanto de **36.580,02€** con una duración estimada de tres meses.

7. CONCLUSIONES, OBJETIVOS CUMPLIDOS, LÍNEAS FUTURAS DE TRABAJO

Tras la elaboración de este documento, el objetivo principal marcado en todo el proyecto ha sido cumplido. A través de la arquitectura planteada de red neuronal convolucional siamesa, junto con el uso de las redes de MobileNet y las técnicas de penalización de pesos en caso de falsos positivos durante la fase de entrenamiento se ha logrado implementar un modelo no realizado en otras publicaciones que permite obtener unos resultados muy positivos.

En cuando a los objetivos parciales, el profundo análisis del estado del arte ha permitido entender y analizar las arquitecturas más utilizadas en cuando a verificación biométrica, conociendo el rendimiento y resultados obtenidos por cada una de ellas, además de conocer lo que realmente se está utilizando en este trabajo (Arquitecturas de Inception-ResNet, MobileNet, etc.)

Por otro lado, se ha podido diseñar e implementar toda la red siamesa de manera correcta, utilizando las redes de Inception-ResNet e MobileNet como redes del Bloque I del modelo, y un clasificador binario como parte del bloque II. Con ello, y a través de la penalización de falsos positivos a través de un aumento de la pérdida o *loss* en estos casos, se han obtenido unos resultados muy positivos en cuando a la exactitud del modelo, y una tasa de falsos positivos o posibles intrusos prácticamente nula.

Por último, el objetivo parcial restante ha sido correctamente cumplido gracias a la implementación de las redes de MobileNet, mucho más ligeras y veloces que las redes Inception-ResNet, y con unos resultados en cuanto a tiempos de ejecución muy atractivos en este tipo de problemas.

Con todo ello, se ha podido comprobar y cumplir los objetivos propuestos de este trabajo, donde se ha logrado realizar una propuesta de mejora en la aplicación de técnicas de verificación facial en dispositivos móviles, basada en los resultados obtenidos de las arquitecturas pre-entrenadas propuestas de redes de neuronas convolucionales como MobileNet e Inception-ResNet. Además, existen posibles ramas de fututo en las que poder continuar con la resolución de este problema de biometría facial:

- Con el fin de encontrar una posible mejora de los resultados futuros, a partir de este trabajo es posible continuarlo a través del desarrollo de técnicas como la posible combinación de inputs de datos basados en imágenes de personas además de audio o voz de las mismas. Una posible solución que podría permitir aumentar la seguridad ante intrusos, una vez realizaras las pruebas necesarias.
- Existe la posibilidad además de implementar otros tipos de clasificadores supervisados como alguno de los siguientes, analizando cada uno de los resultados y comparándolos para encontrar una posible alternativa que logre mejorar las prestaciones del modelo:
    - Máquinas de Vectores de Soporte ("*Support Vector Machines*").
    - Arboles de Decisión ("*Decision Tree*").
    - Bosques Aleatorios ("*Random forest*").
    - Conjunto de clasificadores ("*Ensemble Classifier*").
    - Regresión Logística ("*Logistic Regression*").

    - Regresión Lasso ("*Lasso Regression*").
    - Bayes ingenuo ("*Naive Bayes*").
- El campo de la biometría facial está en auge estos últimos años, y por eso existen además nuevas arquitecturas descubiertas recientemente que podrían permitir aumentar la precisión y resultados del modelo sin necesidad de aumentar el tiempo de ejecución. TAN y LE [56] mostraron al público el potencial de las redes EfficientNet, que permiten adaptarse a las necesidades específicas en función de los parámetros de entrada para utilizar una arquitectura de la red distinta en cada situación. Estas redes pueden ser de gran interés para un posible estudio futuro basado en este mismo.
- También existe la posibilidad de crear un modelo que se capaz de utilizar este mismo para realizar verificaciones por medio de procesamiento de video automático, en tiempo real, existiendo la posibilidad de mejorar sus resultados tras la combinación de las imágenes de cada video.

En definitiva, la elaboración de este documento ha logrado cumplir con los objetivos planteados, permitiendo dar paso a una línea potencialmente aplicable en muchos dispositivos actuales que requieren de tecnologías de verificación facial con unos recursos computacionales limitados, como pueden ser teléfonos móviles, cajeros automáticos, cámaras de video-vigilancia privada o de lugares de alta seguridad como aeropuertos, etc.

8. REFERENCIAS

[1] J. Woodward, C. Horn, J. Gatune y A. Thomas, «Biometrics: A Look at Facial Recognition.,» 2003. [En línea]. Available: https://apps.dtic.mil/docs/citations/ADA414520. [Último acceso: 2019 Septiembre 15].

[2] C. Blakemore, «Bertillon System,» Encyclopedia.com, 2019. [En línea]. Available: https://www.encyclopedia.com/social-sciences-and-law/law/crime-and-law-enforcement/bertillon-system.

[3] A. Alejo Combarro, R. Larin-Fonseca y H. Vazquez, 3D Face Models Reconstruction based on bi-dimensional images, 2014.

[4] J.-S. Pierrard y T. Vetter, Skin Detail Analysis for Face Recognition, 2007.

[5] J. Wolfe y J. Dastin, «U.S. government study finds racial bias in facial recognition tools.,» U.S., 2019. [En línea]. Available: https://www.reuters.com/article/us-usa-crime-face/u-s-government-study-finds-racial-bias-in-facial-recognition-tools-idUSKBN1YN2V1.

[6] D. THAKKAR, «Bayometric.com. Biometric Authentication Now and Then: History and Timeline.,» 2019. [En línea]. Available: https://www.bayometric.com/biometric-authentication-history-timeline/. [Último acceso: 2019 Octubre 1].

[7] M. Turk y A. Pentland, «Eigenfaces for Recognition,» de *Journal of Cognitive Neuroscience*, 1991, pp. 3(1), pp.71-86.

[8] M. Savvides, B. V. Kumar y P. Khosla, Face verification using correlation filters, 2002.

[9] Y. Wang, T. Tan y A. K. Jain, Combining face and iris biometrics for identity verification. In International Conference on Audio-and Video-Based Biometric Person Authentication (pp. 805-813), Springer, Berlin, Heidelberg, 2003.

[10] N. &. C. R. Ramanathan, «Face verification across age progression.,» de *IEEE Transactions on Image Processing, 15(11),*, 2006, pp. 3349-3361..

[11] M. Zhou y H. Wei, «Face verification using gaborwavelets and adaboost. In 18th International Conference on Pattern Recognition (ICPR'06) (Vol. 1, pp. 404-407). IEEE.,» 2006. [En línea].

[12] N. Kumar, A. C. Berg, P. N. Belhumeur y S. K. Nayar, Attribute and simile classifiers for face verification. In 2009 IEEE 12th International Conference on Computer Vision (pp. 365-372). IEEE., 2009.

[13] F. ROSENBLATT, «Perceptron simulation experiments,» de *PROCEEDINGS OF THE INSTITUTE OF RADIO ENGINEERS*, 1959, pp. 468-468.

[14] A. Cartas, «Wikipedia,» 12 07 2015. [En línea]. Available: https://es.wikipedia.org/wiki/Perceptr%C3%B3n#/media/Archivo:Perceptr%C3%B3n_5_unidades.svg.

[15] 3Blue1Brown, «What is backpropagation really doing? | Deep learning, chapter 3,» 3 11 2017. [En línea]. Available: https://www.youtube.com/watch?v=Ilg3gGewQ5U.

[16] S. Shehata, «Using mid- and high-level visual features for surgical workflow detection in cholecystectomy procedures,» 2016.

[17] D. H. Hubel y T. N. Wiesel, Receptive fields and functional architecture of monkey striate cortex. The Journal of physiology, 195(1), 215-243., 1968.

[18] K. Chellapilla, S. Puri y P. Simard, High performance convolutional neural networks for document processing., 2006.

[19] A. Berg, J. Deng y L. Fei-Fei, Large scale visual recognition challenge (ILSVRC)., 2010.

[20] S. Chopra, R. Hadsell y Y. LeCun, Learning a similarity metric discriminatively, with application to face verification. In CVPR (1) (pp. 539-546)., 2005.

[21] L. DENG, The mnist database of handwritten digit images for machine learning research, 6 ed., vol. 29, IEEE Signal Processing Magazine, 2012, pp. 141-142.

[22] L. Fei-Fei, «ImageNet: crowdsourcing, benchmarking & other cool things,» CMU VASC Seminar, Marzo 2010. [En línea]. Available: http://image-net.org/about-publication.

[23] N. Kumar, A. C. Berg, P. N. Belhumeur y S. K. Nayar, «International Conference on Computer Vision (ICCV),» 2009.

[24] G. B. Huang, M. Ramesh, T. Berg y E. Learned-Miller, «Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments.,» University of Massachusetts, Amherst, 2007.

[25] Y. Sun, X. Wang y X. Tang, «Deep learning face representation from predicting 10,000 classes.,» CVPR, 2014.

[26] J. e. a. HU, «Large margin multi-metric learning for face and kinship verification in the wild.,» de *Asian conference on computer vision.*, Springer, Cham, 2014., pp. 252-267.

[27] Y. e. a. SUN, «Deep learning face representation by joint identification-verification.,» de *Advances in neural information processing systems.*, 2014, pp. 1988-1996.

[28] J. e. a. ZHOU, «Biometric Recognition: 12th Chinese Conference,» Shenzhen, China, Springer, 2017.

[29] I. Masi, Y. Wu, T. Hassner y P. Natarajan, « Deep face recognition: A survey.31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI) (pp. 471-478). IEEE.,» 2018.

[30] A. Srivastava, «Label Smoothing: Making model robust to incorrect labels.,» 5 Marzo 2019. [En línea]. Available: https://towardsdatascience.com/label-smoothing-making-model-robust-to-incorrect-labels-2fae037ffbd0. [Último acceso: 2020].

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov y A. Rabinovich, Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9)., 2015.

[32] K. He, X. Zhang, S. Ren y J. Sun, Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778)., 2016.

[33] B. Raj, «A simple Guide to the versions of the Inception network.,» 2018. [En línea]. Available: https://towardsdatascience. com/a-simpleguide-to-the-versions-of-the-inception-network-7fc52b863202..

[34] S.-H. Tsang, «Review: Inception-v4 — Evolved From GoogLeNet, Merged with ResNet Idea (Image Classification),» Medium, 27 Septiembre 2018. [En línea]. Available: https://towardsdatascience.com/review-inception-v4-evolved-from-googlenet-merged-with-resnet-idea-image-classification-5e8c339d18bc. [Último acceso: 8 Junio 2020].

[35] S. IOFFE y C. SZEGEDY, Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167, 2015..

[36] M. DelSole, «What is One Hot Encoding and How to Do It,» Medium, 25 Abril 2018. [En línea]. Available: https://medium.com/@michaeldelsole/what-is-one-hot-encoding-and-how-to-do-it-f0ae272f1179.

[37] C. Szegedy, S. Ioffe, V. Vanhoucke y A. A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning. In Thirty-First AAAI Conference on Artificial Intelligence., 2017.

[38] A. G. HOWARD, Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861., 2017.

[39] I. Mansouri, «Computer Vision Part 4: An overview of Image Classification architectures,» Medium, 2019. [En línea]. Available: https://medium.com/@ilias_mansouri/part-4-image-classification-9a8bc9310891. [Último acceso: 2020 Junio 08].

[40] M. e. a. M. SANDLER, Inverted residuals and linear bottlenecks. En Proceedings of the IEEE conference on computer vision and pattern recognition. p. 4510-4520., 2018.

[41] D. Han, J. Kim y J. Kim, Deep pyramidal residual networks., 2016.

[42] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le y H. Adam, «Searching for MobileNetV3,» Cornell University, 2019.

[43] J. HU, L. SHEN y G. SUN, «Squeeze-and-excitation networks.,» de *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132-7141.

[44] C. SZEGEDY y e. al., «Inception-v4, inception-resnet and the impact of residual connections on learning.,» 2016. [En línea]. Available: http://arxiv. org/abs/1602.07261.

[45] S. BIANCO y e. al., «Benchmark analysis of representative deep neural network architectures,» vol. 6, 2018, pp. 64270-64277.

[46] A. HOWARD y e. al., «Searching for mobilenetv3.,» de *Proceedings of the IEEE International Conference on Computer Vision.*, 2019, pp. 1314-1324.

[47] Pedregosa y a. et, «Scikit-learn: Machine Learning in Python,» *JMLR 12,* pp. 2825-2830, 2011.

[48] B. MASURKAR, «Supervised vs Unsupervised Machine Learning.,» StepUp Analytics., Octubre 2018. [En línea]. Available: https://stepupanalytics.com/supervised-vs-unsupervised-machine-learning/. [Último acceso: 2017 Abril 2020].

[49] M. Jeeva y M. Jeeva, «The Scuffle Between Two Algorithms -Neural Network vs. Support Vector Machine.,» Medium., 2018. [En línea]. Available: https://medium.com/analytics-vidhya/the-scuffle-between-two-algorithms-neural-network-vs-support-vector-machine-16abe0eb4181 . [Último acceso: 21 Abril 2020].

[50] M. Alvarado, «Image Recognition - Gender Detection - Inceptionv3.,» Kaggle, 2020. [En línea]. Available: https://www.kaggle.com/bmarcos/image-recognition-gender-detection-inceptionv3/#data. [Último acceso: 9 Marzo 2020].

[51] G. Bonaccorso, Mastering Machine Learning Algorithms, Packt Publishing, 2018.

[52] R. Karim, «Intuitions on L1 and L2 Regularisation,» Medium, 2018. [En línea]. Available: https://towardsdatascience.com/intuitions-on-l1-and-l2-regularisation-235f2db4c261. [Último acceso: 12 Marzo 2020].

[53] N. Chinchor, MUC4 '92: Proceedings of the 4th conference on Message understanding, McLean, Virginia: {Association for Computational Linguistics},, 1992.

[54] J. Brownlee, «A Gentle Introduction to the Fbeta-Measure for Machine Learning. Machine Learning Mastery.,» 13 Mayo 2020. [En línea]. Available: https://machinelearningmastery.com/fbeta-measure-for-machine-learning/ .

[55] T. Lasko, J. Bhagwat, K. Zou y L. Ohno-Machado, «The use of receiver operating characteristic curves in biomedical informatics.,» Journal of Biomedical Informatics, 2005, pp. 404-415.

[56] M. TAN y Q. V. LE, «Efficientnet: Rethinking model scaling for convolutional neural networks.,» arXiv, 2019.

[57] C. J. VAN RIJSBERGEN, «Retrieval effectiveness. Information retrieval experiment.,» 1981, pp. 32-43..

[58] F. SCHROFF, D. KALENICHENKO y J. PHILBIN, «Facenet: A unified embedding for face recognition and clustering. En Proceedings of the IEEE conference on computer vision and pattern recognition.,» 2015, pp. 815-823.

[59] Y. SASAKI, «The Truth of the F-Measure.,» 2007.

[60] T. A. PIETRASZEK. Tadeusz, «Data mining and machine learning—towards reducing false positives in intrusion detection. Information security technical report,» vol. 10, 2005, pp. 169-183.

[61] A. Krizhevsky, I. Sutskever y G. Hinton, Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105)., 2012.

[62] Juanes y G. G., «Cuadernos de Seguridad,» 2018. [En línea]. Available: https://cuadernosdeseguridad.com/2018/11/aplicaciones-de-la-biometria-en-la-vida-cotidiana/. [Último acceso: 16 Septiembre 2019].

[63] H. Huttunen, «SGN-41007 Pattern Recognition and Machine Learning. Cs.tut.fi.,» 2019. [En línea]. Available: http://www.cs.tut.fi/courses/SGN-41007/. [Último acceso: 2019 Septiembre 30].

[64] G. B. Huang, M. Ramesh, T. Berg y E. Learned-miller., Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In ECCV Workshop on Faces in Real-life Images,, 2008.

[65] D. Georgescu, University of Economic Studies, Bucharest, 2011.

[66] S. F., «La biometría para la identificación de las personas,» 2015. [En línea]. Available: https://www.academia.edu/31531606/La_biometr%C3%ADa_para_la_identificaci%C3%B3n_de_las_personas_Francesc_Serratosa_PID_00195448.

[67] ComputerHoy, «Reconocimiento facial: ventajas y peligros de una revolución.,» [En línea]. Available: https://computerhoy.com/noticias/life/reconocimiento-facial-ventajas-peligros-revolucion-69441. [Último acceso: 2019].

[68] Z. Bukovčiková, D. Sopiak, M. Oravec y J. Pavlovičová, Face verification using convolutional neural networks with Siamese architecture. In 2017 International Symposium ELMAR (pp. 205-208). IEEE., 2017.

[69] A. Pérez Silverio, «Datos biométricos ¿Qué y cuáles son? ¿Cómo cumplir con la Ley?,» 2020. [En línea]. Available: https://ayudaleyprotecciondatos.es/2019/02/15/datos-biometricos/.

[70] M. Jeeva, «The Scuffle Between Two Algorithms -Neural Network vs. Support Vector Machine.,» Medium, 2018. [En línea]. Available: https://medium.com/analytics-vidhya/the-scuffle-between-two-algorithms-neural-network-vs-support-vector-machine-16abe0eb4181. [Último acceso: 21 Abril 2020].

[71] L. Zhang, G. Gui, A. M. Khattak, M. Wang, W. Gao y J. Jia, «Multi-task cascaded convolutional networks based intelligent fruit detection for designing automated robot.,» vol. 7, 2019, pp. 56028-56038.

[72] R. Gandhi, «Siamese Network & Triplet Loss,» towardsdatascience, 2018. [En línea].

9. GLOSARIO DE ÁCRONIMOS

2D: Dos Dimensiones

ACC: Exactitud (del inglés *Accuracy*)

AUC: Área Bajo la Curva (del inglés *Area Under the Curve*)

CE: Constitución Española

CNN: Red de Neuronas Convolucional (del inglés *Convolutional Neuronal Network*)

ET: Tiempo de Evaluación

FN: Falsos Negativos (del inglés *False Negatives*)

FNR o **TFN**: Tasa de Falsos Negativos (del inglés *False Negative Ratio*)

FP: Falsos Positivos (del inglés *False Positives*)

FPR o **TFP**: Tasa de Falsos Positivos (del inglés *False Positive Ratio*)

LFW: Caras Etiquetadas en la Nube (del inglés *Labeled Faces in the Wild*)

LO: Ley Orgánica

ML: Machine Learning

RFID: IDentificación por RadioFrecuencia (del inglés *Radio Frequency Identification*)

RGPD: Reglamento General de Protección de Datos

ROC: curva de Característica Operativa del Recepto (del inglés *Receiver Operating Characteristic Curve)*

SVM: Máquina de Vectores de Soporte (del inglés *Support Vector Machine*)

TF: TensorFlow

TN o **VN**: Verdaderos Negativos (del inglés *True Negatives*)

TNR o **TVN**: Tasa de Verdaderos Negativos (del inglés *True Negative Ratio*)

TP o **VP**: Verdaderos Positivos (del inglés *True Positives*)

TPR o **TVP**: Tasa de Verdaderos Positivos (del inglés *True Positive Ratio*)

TT: Tiempo de Entrenamiento

ANEXO: PROCESO DE APLICACIÓN BIOMÉTRICA

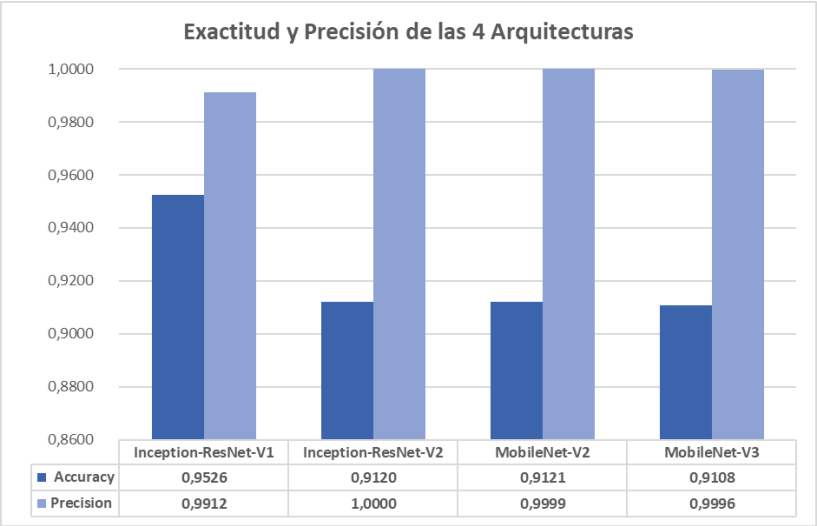
En este apartado se propone una implementación de la tecnología estudiada en el trabajo para poder ser utilizada en un sistema de verificación facial en un dispositivo móvil.

La elaboración de este sistema es simplemente teórica en un primer supuesto, aplicando los conocimientos adquiridos a lo largo de todo el trabajo.

Para ello, se han aprovechado los resultados positivos que han mostrado tanto el diseño de la red siamesa como la aplicación de la red de MobileNet, al ser más ligera y veloz y con unos resultados similares al resto de arquitecturas.

Para el caso de uso, se ha considerado un entorno de dispositivo móvil, portable, o sistemas empotrados limitados, donde existe una alta prioridad en los tiempos de ejecución de cada resultado.

Recordando los resultados de MobileNetV3 se pueden ver las tasas de errores, que son a su vez las probabilidades de fallar en cada ejecución:



Los resultados de MobileNetV3 has sido de una exactitud de 0,9108, y una precisión de 0,9996. A través de estos datos, se deduce que la tasa de falsos positivos (TFP) es igual a la diferencia de la unidad frente a la precisión, y la tasa de fallos totales (TFP + TFN) es igual a la diferencia de la unidad frente a la exactitud. Por tanto, ambos valores son de:

Ratio	Porcentaje
Tasa de Falsos Positivos	0.04%
Tasa de Fallos	8.92%

Estos valores son solamente los establecidos para una sola predicción o comparación “1:1” entre una imagen guardada previamente en una base de datos y la imagen a verificar. Para este caso de uso, aprovechando las rápidas tasas de inferencias que posee MobileNetV3, se pretende realizar una predicción de la imagen a verificar por cada una de las imágenes que se han guardado anteriormente de la persona a identificar, estableciendo una relación “1:N”. Con ello, se logra aumentar la exactitud promedia al permitir al sistema poder comparar la imagen con varias anteriores, y obtener conclusiones a partir de todas ellas.

Para este caso de uso, se propone la posible utilización del modelo para realizar varias inferencias. El proceso es el siguiente:

1. Se realiza un proceso único previo de registro de la persona a verificar, realizando varias fotos de la misma y obteniendo los cuellos de botella o “bottlenecks” del Bloque I, por cada una de ellas.
2. Por cada imagen que vaya a ser comparada con las almacenadas previamente (los “bottlenecks” previos) se realiza lo siguiente:
    a. **Fase de detección**: se recoge el rostro de la imagen realizada, ya sea a partir de una foto o un vídeo.
    b. **Pre-procesamiento**: se realizan las operaciones previas al modelo, como pueden ser la alineación del rostro, ajustes de luminosidad, etc.
    c. **Obtención del “bottleneck”** de la nueva imagen a comparar. En esta fase se realiza la inferencia del bloque I para obtener su cuello de botella previo al Bloque II.
    d. **Clasificación binaria** bloque II. se realiza una inferencia o predicción entre el bottleneck de la nueva imagen y los bottlenecks guardados previamente. Se obtiene un resultado por cada comparación 1 a 1.
    e. **Toma de decisiones** de las predicciones. A partir de los resultados de las predicciones realizadas (una por cada imagen almacenada previamente), se procede a establecer una decisión final basada en todas ellas. Esta política de decisión variará en función del nivel de restricción que se establezca dependiendo de la situación:
        i. Para situaciones de alta seguridad como pueden ser la autenticación de individuos en el propio teléfono, se puede establecer una proporción mínima de predicciones positivas (se considera la misma persona) mayor al 90% o incluso el 100%. Con ello se logra una gran reducción de posibles intrusos (FP), pero aumenta en contrapartida los fallos al intentar acceder una persona autorizada (FN).
        ii. Para situaciones que requieran menor seguridad, esta restricción se puede modificar para reducir las denegaciones totales. Por ejemplo, para un caso donde se pretenda encontrar imágenes de la galería de fotos donde aparezca la persona en cuestión, se deberá reducir esta restricción de manera que se puedan encontrar todas las imágenes posibles de esta persona, en lugar de solamente unas pocas imágenes que sean muy probables de que sean este individuo.

3. A partir de la fase anterior, se puede realizar el mismo proceso tantas veces como imágenes a verificar se vayan a utilizar. De esta manera, se podrá establecer una restricción muy elevada en las predicciones de cada imagen, y comenzar a realizar verificaciones por cada imagen nueva a autorizar, estableciendo un máximo número de nuevas imágenes, e ir probando hasta que se pueda encontrar alguna con una predicción positiva. De esta manera el sistema se asegura de:
    a. Realizar tantas verificaciones como sean necesarias, flexible en cada situación.
    b. Asegurar un nivel de seguridad mínimo para cada verificación. Cada imagen se comparará no solo con una anterior almacenada, sino con todas las que se hayan considerado como aptas para realizar las inferencias, e incluyendo la restricción de todas estas predicciones hayan sido tomadas como positivas

A continuación, se muestra el diagrama de flujo planteado para este tipo de diseño en el caso de uso concreto:

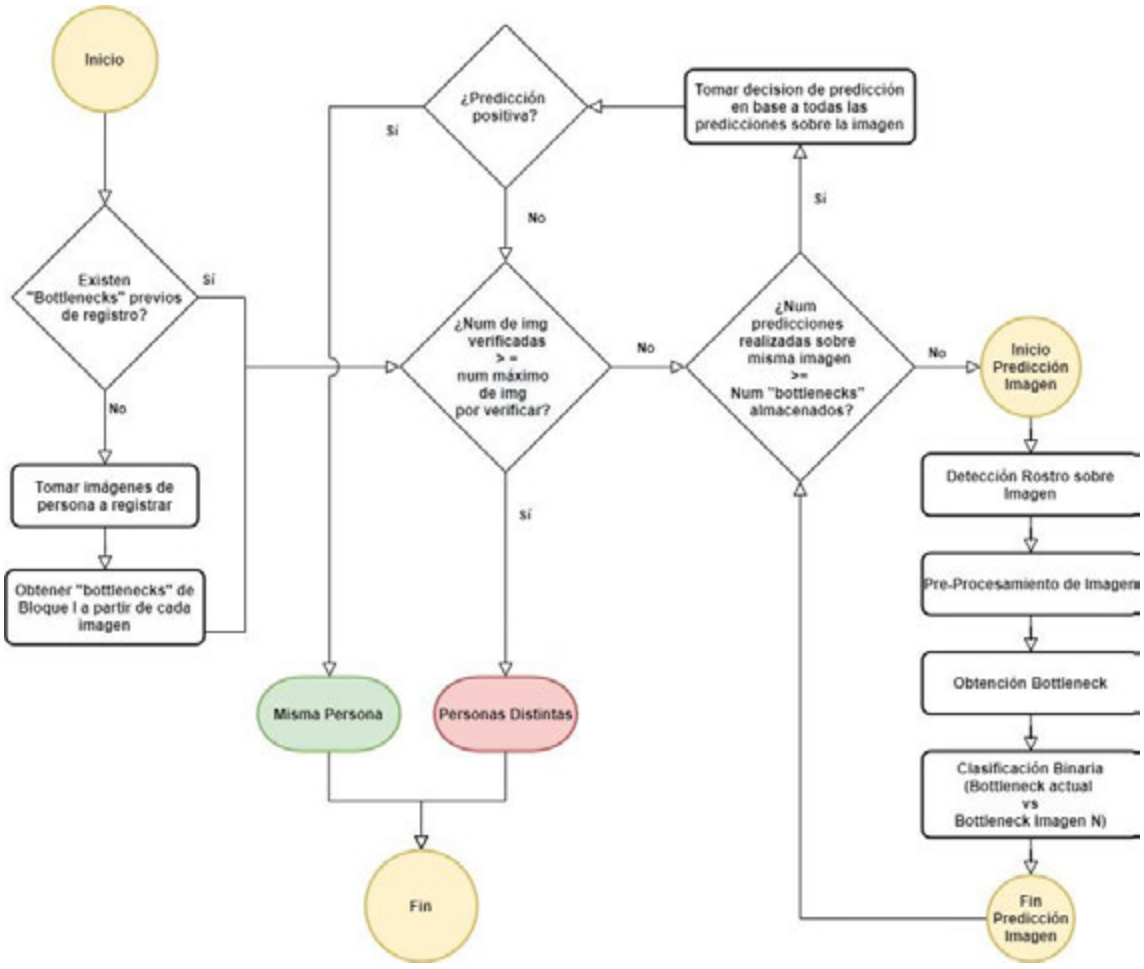


Ilustración 50: Diagrama de flujo planteado sobre el caso de uso: autenticación facial en dispositivos móviles.

ANEXO FINAL: DECLARACION DE ORIGINALIDAD

Yo, Asier Alcaide Martinez, declaro que el TFG "Redes Neuronales Convolucionales Siamesas aplicadas a la Verificación Facial " es totalmente original mío, que no ha sido presentado en ninguna otra universidad como TFG y que todas las fuentes que han sido utilizadas han sido adecuadamente citadas y aparecen en las referencias bibiográficas.

Colmenarejo, a 6 de Julio de 2020

Firma: