Universidad Europea

UNIVERSIDAD EUROPEA DE MADRID

ESCUELA DE ARQUITECTURA, INGENIERÍA Y DISEÑO

MÁSTER UNIVERSITARIO EN

ANÁLISIS DE DATOS MASIVOS (BIG DATA)

TRABAJO FIN DE MÁSTER

CLASIFICACIÓN DE HUELLAS DACTILARES MEDIANTE REDES NEURONALES CONVOLUCIONALES

NOMBRE:

LAURA OLIVA BERENGUER

CURSO 2021-2022

Universidad Europea

TÍTULO: Clasificación de huellas dactilares mediante Redes Neuronales Convolucionales

AUTOR: Laura Oliva Berenguer

TITULACIÓN: MÁSTER EN ANÁLISIS MASIVO DE DATOS

DIRECTOR DEL PROYECTO: Jorge Luis Hita

FECHA: Octubre de 2022

RESUMEN

El mundo de las ciencias forenses no deja de crecer, cada vez son más las nuevas técnicas y líneas de investigación que se desarrollan para las diferentes disciplinas que engloba la criminalística, pero también los criminales y las nuevas formas de crimen se renuevan y actualizan de manera exponencial.

Una de las disciplinas que forma parte de la criminalística es la lofoscopia, y los criminales encuentran maneras de falsificar las huellas de las personas para cometer actos delictivos. Por ello, es importante conocer bien la morfología de las impresiones de cada ser humano, y saber cómo clasificarlas y utilizarlas para la resolución de los crímenes.

Por otro lado, las redes neuronales están en pleno desarrollo, por lo que entenderlas y dominarlas puede suponer verdaderos avances en nuestra sociedad. Este trabajo pretende aprovechar este crecimiento del Machine Learning para crear y entrenar desde cero un modelo de red neuronal que sea capaz de clasificar imágenes de huellas dactilares en función de su delta, basándose concretamente en el uso de las redes neuronales convolucionales (CNN).

Palabras clave: huellas dactilares, Redes Neuronales Convolucionales, delta, clasificación

ABSTRACT

The world of forensic sciences does not stop growing, more and more new techniques and lines of investigation are being developed for the different disciplines that criminalistics encompasses, but also criminals and new forms of crime are renewed and updated exponentially.

One of the disciplines that are part of criminalistics is lofoscopy, and criminals find ways to forge people's fingerprints to commit criminal acts. Therefore, it is important to have a good knowledge of the morphology of the prints of each human being, and know how to classify and use them to solve crimes.

On the other hand, neural networks are in full development, so understanding and mastering them can mean real advances in our society. This work aims to take advantage of this growth of Machine Learning to create and train from scratch a neural network model that is able to classify fingerprint images based on their delta, specifically based on the use of convolutional neural networks (CNN).

Key words: fingerprints, Convolutional Neural Networks, delta, classification

AGRADECIMIENTOS

En primer lugar, agradecer a mi tutor Jorge Luis Hita por todo su apoyo y su ayuda durante estos meses. Gracias por tus ánimos y por estar siempre dispuesto a dedicarme todo el tiempo necesario.

Gracias también a la Universidad Europea por haberme permitido recibir esta formación y darme la oportunidad de continuar con mi desarrollo personal y profesional.

Por último, gracias a mis padres por estar siempre conmigo.

Índice

Capítulo 1. INTRODUCCIÓN

1.1 Planteamiento del problema

Actualmente en las ciencias forenses, el estudio de las impresiones dactilares supone un campo importante a tener en cuenta en el transcurso de las investigaciones policiales, pues al ser únicas para cada ser humano, permiten la identificación de los sospechosos y/o víctimas.

Un problema recurrente en las investigaciones de crímenes es el gran tiempo que supone la clasificación de estas impresiones o huellas dactilares, debido a la calidad de las mismas o a su complejidad en cuanto a morfología. Además, puede ser que, para determinados casos, el número de huellas a tener en cuenta sea elevado, lo que supone una gran inversión de tiempo. La automatización y el uso de las nuevas tecnologías aplicada en el ámbito de las ciencias forenses puede suponer un gran avance.

Estos avances pueden contemplar el uso del aprendizaje automático o Machine Learning, que permitirá trabajar con un modelo que proporcione una clasificación lo más precisa posible de las huellas dactilares recogidas.

Con este proyecto se plantea el uso de estas redes neuronales para automatizar la clasificación de huellas dactilares y facilitar así la labor de los cuerpos policiales encargados de las investigaciones criminales.

1.2 Objetivos del proyecto

Este trabajo de investigación tiene como objetivo principal construir desde cero una red neuronal que sea capaz de clasificar las huellas dactilares en función de uno de sus puntos característicos llamado delta, el cual se definirá en el siguiente capítulo. Para conseguir esto, se tendrán en cuenta los siguientes subobjetivos:

- Recopilación de una amplia base de datos de huellas dactilares.
- Procesamiento de las imágenes obtenidas para conseguir un mismo formato en el dataset completo.
- Creación de un modelo de inteligencia artificial basado en una red neuronal convolucional que logre clasificar huellas dactilares en función del punto característico escogido.

1.3 Hipótesis del proyecto

Una vez construido el modelo de red neuronal, se pretende obtener el mejor rendimiento empleando las métricas correspondientes.

Dentro del mencionado delta podemos encontrar cuatro clases (Adelto, Bidelto, Dextrodelto y Sinistrodelto) que serán definidas en el siguiente capítulo. Teniendo en cuenta estas clases, se parte de un dataset desbalanceado, contando con un gran número de imágenes para las clases Dextrodelto y

Sinistrodelto, y pocas para Adeltos y Bideltos. Por eso, los resultados obtenidos del modelo serán buenos para las dos primeras clases; sin embargo, para las otras dos clases, la clasificación no contará con un porcentaje de acierto tan alto.

1.4 Estructura del proyecto

El proyecto se estructura en diferentes apartados que se describen a continuación:

- **Introducción**. Contiene un breve desarrollo del planteamiento del problema, así como una primera toma de contacto con los objetivos del proyecto.

- **Memoria técnica**. En este capítulo se recogen varios apartados:

    - **Objetivos**: se desarrollan en mayor profundidad los objetivos que persigue el proyecto
    - **Marco teórico**: se recoge también el marco teórico, con información relativa a la historia de la dactiloscopia y los antecedentes, también se incluye una sección sobre las redes neuronales convolucionales y su funcionamiento.
    - **Metodología**: además se incluye la metodología, donde se detallan los materiales y métodos empleados.
    - **Resultados**: por último, se recogen los resultados obtenidos.

- **Conclusiones**. Se recogen finalmente todas las conclusiones extraídas de los resultados del entrenamiento del modelo.
    - **Conclusiones del modelo**: se extraen las conclusiones sobre todos los modelos elaborados, y unas conclusiones finales sobre el mejor modelo obtenido.
    - **Visualización mediante Power BI**: por último, se concluye con un dashboard elaborado en Power BI que recoge una visión global de los datos empleados y los resultados que se han obtenido.

Capítulo 2. Memoria técnica

2.1 Objetivos

El objetivo principal de este proyecto es la elaboración de un modelo de red neuronal que sea capaz de clasificar las huellas dactilares en función de su delta.

Para conseguir este objetivo, es necesario definir unos objetivos indirectos para la realización del proyecto:

- Conocer y entender la morfología de las impresiones y huellas dactilares
- Clasificar las huellas dactilares en función del punto característico llamado delta y diferenciar a simple vista las distintas clases
- Conocer los diferentes tipos de redes neuronales y escoger el modelo más adecuado para el propósito del proyecto

Además, se definen objetivos directos y específicos:

- Recopilación de una amplia base de datos de huellas dactilares, tanto recogidas en papel a voluntarios, como obtenidas de bases de datos ya existentes.
- Procesamiento de las imágenes obtenidas para conseguir un mismo formato en el dataset completo.
- Creación de un modelo de inteligencia artificial basado en una red neuronal convolucional que logre clasificar huellas dactilares en función de su delta.

2.2 Marco teórico

2.1.1 Lofoscopia

El estudio de las huellas dactilares y su importancia crecen exponencialmente. Si hay una buena forma de identificar a las personas, esa es empleando las impresiones dactilares. No solo se estudian las huellas dactilares, sino cualquier patrón generado por las crestas encontrado en las palmas de las manos y las plantas de los pies.

Pero, ¿por qué es tan importante su estudio? Gracias a ellas, se puede conocer la identidad de los criminales, al recoger las huellas halladas en la escena de delito, lo cual puede suponer un indicio adicional a la hora de esclarecer los hechos junto al resto de las pruebas halladas. Además, también resulta interesante su estudio en el proceso de levantamiento de cadáveres, ya que no es solo útil conocer la identidad de los delincuentes, sino también de las víctimas, permitiendo el cotejo del registro policial con las huellas tomadas de los fallecidos [1].

De esto se encarga la lofoscopia; dicha palabra tiene su origen en *lofos*, que significa cresta, relieve, y *skopein*, que significa observar. Por tanto, se puede

definir la lofoscopia como la ciencia encargada de estudiar los dibujos creados por las crestas papilares en manos y pies.

Pasando por la quiroscopia (estudio de las palmas de las manos) y la pelmatoscopia (estudio de las plantas de los pies), este trabajo se ha centrado en la dactiloscopia, que se encarga únicamente del estudio de las crestas de los dedos.

Si se centra la vista en sus inicios, las impresiones dactilares ya eran utilizadas cientos de años atrás, como por ejemplo durante el año 701 en la antiguas Leyes de Taiho[1] en Japón, donde no solo se firmaban los documentos de forma manuscrita, sino que también se utilizaba la huella dactilar como manera de sellar los documentos [2].

A pesar de la notable presencia de las huellas dactilares, su fin no era la identificación única de las personas, sino que era un mero uso decorativo o para dejar constancia documentacional.

Es ya en el siglo XIX cuando el antropólogo Juan Vucetich comienza a darle esta visión identificativa al uso de las impresiones dactilares, analizando las diez impresiones dactilares de diferentes individuos y concluyendo que, debido a su evidente clasificación en diferentes grupos, podían servir para identificar de forma única a cada ser humano [4]. De esta forma, las investigaciones policiales empezaron a basar la identificación de criminales mediante la comparación de huellas dactilares de los sospechosos con las huellas tomadas de la escena del crimen.

Las huellas, ya sean dactilares, palmares o plantares, se componen de crestas capilares, donde a su vez se localizan los poros. Estas crestas presentan un conjunto de características que permiten conseguir una buena identificación:

- Son **perennes**, es decir, el número de crestas, su posición y dirección no varían a lo largo de la vida.
- Son **inmutables**; cuando se produce algún tipo de alteración en la dermis las crestas vuelven a su dibujo original.
- Son **diversiformes**, esto es, existen tantas formas y variedad de dibujos que no existen dos personas con el mismo patrón.
- Son **imprimibles**, gracias a la altura de las crestas, que permite tomar las impresiones.
- Se **clasifican fácilmente**, debido a su apreciable morfología.

En cuanto al sistema de clasificación de huellas dactilares, el primer modelo establecido y sobre el que se asientan las bases de la clasificación actual fue

[1] Las Leyes de Taiho eran una forma reorganización administrativa de Japón en el año 701, formado por un conjunto de leyes históricas basadas en el confucionismo [3]

desarrollado por Francis Galton, quien determinó tres tipos de huella según su morfología: arco, presilla y verticilo [5] [Figura 1].

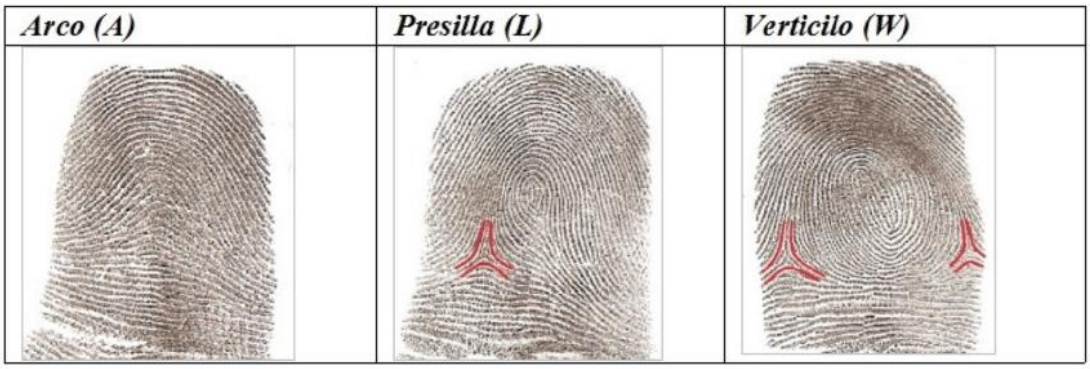


Figura 1. Imagen ejemplo de arco, presilla y verticilo [6]

Este sistema de clasificación evolucionó de la mano del ya mencionado antropólogo Juan Vucetich, quien determinó para la presilla dos subtipos: presilla interna y presilla externa. Este nuevo modelo de clasificación será el utilizado actualmente a la hora de clasificar las huellas dactilares teniendo en cuenta la ubicación del delta, punto característico sobre el que se centra el presente proyecto.

El delta es la forma triangular que puede aparecer de diferentes formas y número en las huellas dactilares, y que se caracteriza por determinar los tres puntos de partida de las crestas que crecen a lo largo de todo el dedo [7]. En la Figura 2 pueden verse ejemplos de deltas.

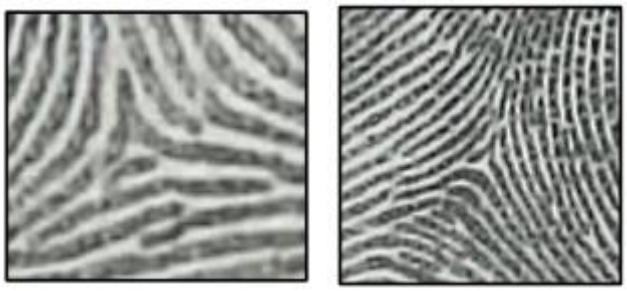


Figura 2. Fuente propia. Ejemplos de deltas

Según las características que presente, se pueden clasificar las huellas dactilares en cuatro tipos [8]:

- **Adeltos**: carecen de delta. En el modelo de Vucetich, se corresponden con el tipo arco [Figura 3 A].
- **Dextrodeltos**: presentan un solo delta, que se localiza a la derecha del núcleo. En el modelo de clasificación de Juan Vucetich, se corresponden con la presilla interna [Figura 3 B].
- **Sinistrodeltos**: presentan un solo delta, localizado a la izquierda del núcleo. Se corresponden con la presilla externa [Figura 3 C].

- **Bideltos**: presentan dos o más deltas. Este último tipo se corresponde con el tipo verticilo [Figura 3 D].

[A]  [B] 

[C]  [D] 

Figura 3. [A] Adelto, [B] Dextrodelto, [C] Sinistrodelto, [D] Bidelto

2.1.2 Aprendizaje profundo. Redes neuronales convolucionales

Como se mencionaba anteriormente, como consecuencia del estado de las huellas latentes recogidas de la escena del crimen, la identificación y comparación de las mismas pueden resultar bastante tediosas para el ojo humano, además de un enorme gasto de tiempo, del cual no se dispone en las investigaciones criminales, ya que cada minuto cuenta.

Es por eso por lo que las nuevas tecnologías suponen una gran ventaja. Si hablamos de clasificación de imágenes, la mejor manera de hacerlo de foma automática es utilizando redes neuronales convolucionales, uno de los algoritmos que ofrece el Deep Learning. Estas redes basadas en el aprendizaje supervisado son la mejor opción, ya que son capaces de identificar curvas, líneas y formas.

Un estudio realizado por la Universidad de Westminster [10], Londres, diseñó un algoritmo empleando redes neuronales convolucionales para la clasificación e identificación de bifurcaciones y abruptas en imágenes de huellas dactilares [Figura 4].

Figura 4. Ejemplo de bifurcación y abrupta [9]

Tras realizar un exhaustivo procesamiento de las imágenes con el fin de visualizar mejor las crestas de las huellas, el algoritmo es capaz de detectar y marcar en la imagen dónde se encuentran los puntos sujeto de estudio [Figura 5] [10].

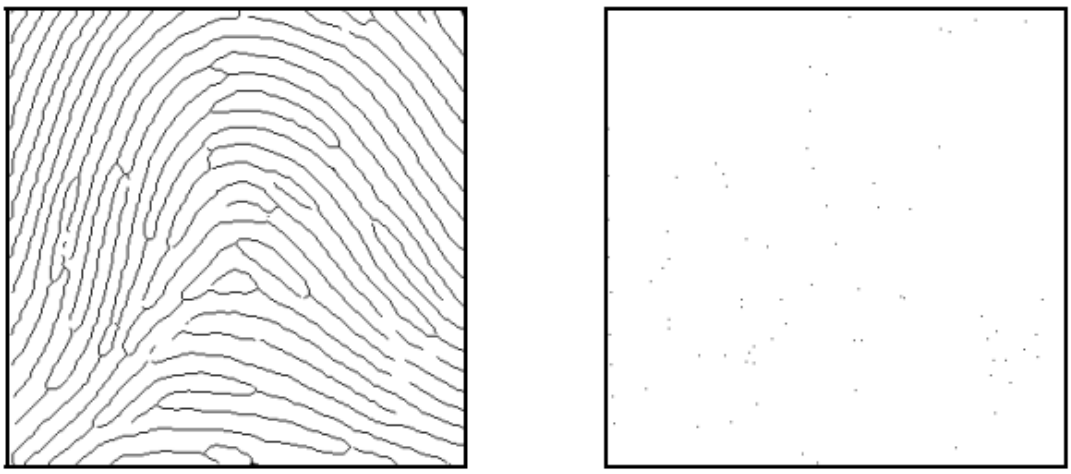


Figura 5

Como se mencionaba anteriormente, el mejor tipo de red neuronal para la clasificación de imágenes es la convolucional, y esto se debe a que utilizan un procesamiento relativamente pequeño. Aprenden de las imágenes de entrada con las que entrenan, y aplicando filtros sobre los píxeles son capaces de identificar patrones [11].

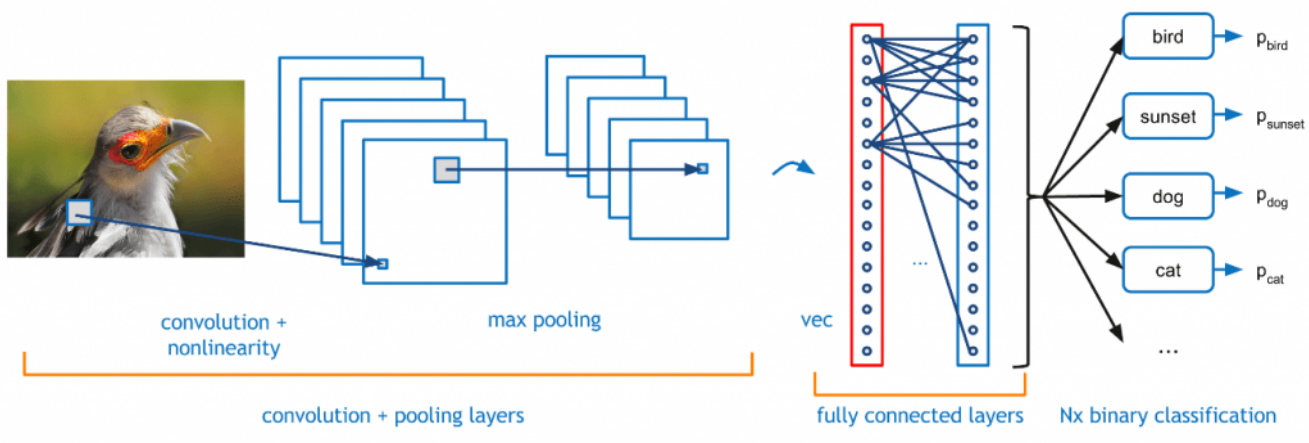


Figura 6. Estructura de red neuronal convolucional [11]

La forma en la que aplica estos filtros es asignando pesos o importancia a determinados objetivos y detalles de la imagen que sean característicos y que permitan así una identificación respecto a los tipos restantes [12].

La estructura de una red convolucional es la siguiente:

- **Capa de entrada**. Recibe el input, es decir, la imagen, y trabaja con ella mediante píxeles. Las neuronas con las que cuenta esta capa es igual al producto de los tres canales (alto x ancho x color) [13].
- **Capas ocultas**. Se encargan de conectar los datos de entrada con la capa de salida para emitir un resultado, transfiriendo la información proporcionada y realizando una serie de operaciones a través de diferentes filtros que se van aplicando. Existen diferentes tipos de capas: convolucionales, *pooling*, *flatten*, *dropout*, etc., que se explicarán en el apartado 2.3.2.2.3 Modelo de red neuronal convolucional.
- **Capa de salida**. Esta última capa se encarga de emitir los resultados del entrenamiento de la red. El número de neuronas que la componga será el total de clases posibles.

Una vez que se han obtenido y preprocesado las imágenes, comienzan a aplicarse las convoluciones o filtros. Estos filtros tendrán un determinado tamaño, e irán recorriendo las imágenes por secciones del tamaño del filtro [Figura 7].

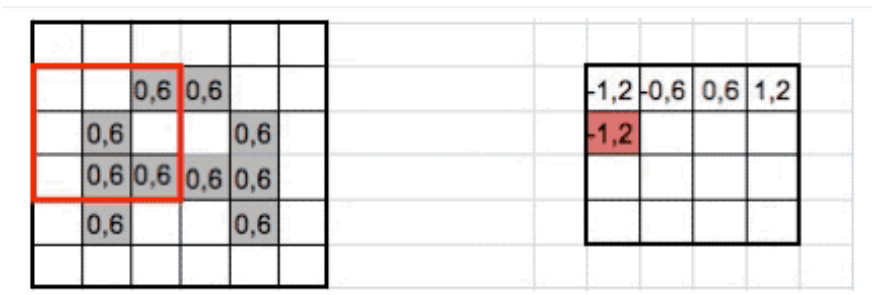


Figura 7. Ejemplo de aplicación de filtro a una imagen [14]

De esta manera, se obtiene una nueva matriz de píxeles; se obtendrán tantas nuevas matrices como número de filtros se hayan especificado.

Además de esto, deberemos realizar un sub-muestreo, ya que de seguir aplicando capas de convolución, el número de neuronas de la siguiente capa crecerá exponencialmente. Es por eso que principalmente se utilizará la técnica de Max Pooling, donde se recorrerán las matrices obtenidas al aplicar la convolución previamente para así reducir su tamaño y eliminando todos los valores excepto los más altos [14].

En la Figura 8 puede verse un esquema de la estructura que siguen las redes neuronales convolucionales.

Una vez comprendida la estructura completa de las redes convolucionales, el modelo irá creciendo y será modificado, añadiéndose un mayor número de capas y fitros, hasta encontrar la configuración que más se adapte al problema planteado.

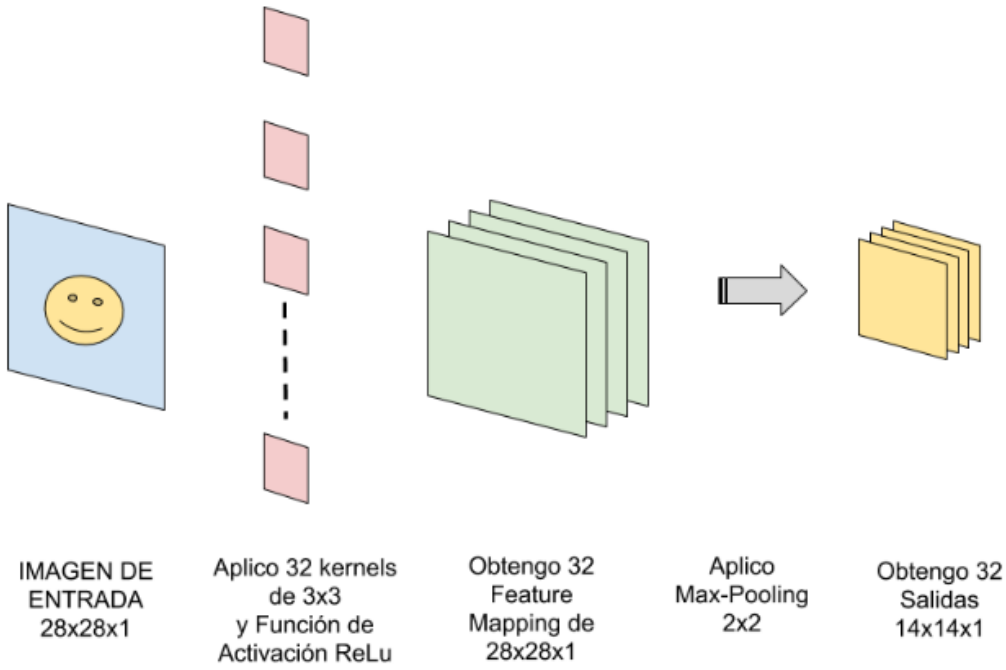


Figura 8. Ejemplo de red convolucional [14]

2.3 Metodología

2.3.1 Materiales

Para la realización del proyecto, se necesitará el siguiente software:

- **Anaconda Navigator**. Es una interfaz gráfica de usuario que permite utilizar diferentes aplicaciones; en este caso es utilizada para programar en lenguaje Python mediante Jupyter Notebook.
- **Power BI**. Herramienta de visualización que permite realizar una representación gráfica de los resultados obtenidos del entrenamiento del modelo.

Además, también será necesaria una base de datos constituida por imágenes de huellas dactilares, procedentes de dos fuentes:

- El 95% de las imágenes han sido tomadas con grafito y papel, y posteriormente escaneadas y transformadas a formato digital.
- El 5% restante de las imágenes se han obtenido de una base de datos del FVC2000, la primera Competición Internacional de Algoritmos de Verificación de huellas dactilares [15].

2.3.2 Métodos

A continuación, se detallan todos los pasos y procedimientos que se llevan a cabo para completar el proyecto.

2.3.2.1 Procesamiento de las imágenes

2.3.2.1.1 Anonimización de las huellas

Para la realización de este proyecto, todas las huellas empleadas han sido anonimizadas, etiquetándose de manera que permitan ser ordenadas pero que no puedan ser asociadas con ningún individuo.

El tratamiento de datos biométricos, como son las huellas dactilares, viene regulado por el Reglamento General de Protección de Datos (RGPD). El artículo 14 de esta ley define que son “datos personales obtenidos a partir de un tratamiento técnico especifico, relativos a las características físicas, fisiológicas o conductuales de una persona física que permitan o confirmen la identificación única de dicha persona, como imágenes faciales o datos dactiloscópicos”. Además, los datos biométricos al ser empleados para la identificación de las personas, deben ser especialmente protegidos [16].

Por otro lado, la regulación sobre la anonimización de las huellas dactilares también viene recogida en la Ley Orgánica de Protección de Datos (LOPD). Menciona que si las huellas o cualquier dato biométrico hace referencia a ideologías, orientación sexual o salud de las personas, solo se podrá acceder a ellas si el propietario así lo autoriza [16].

Por ello, siendo el fin del proyecto la clasificación de huellas pero no su identificación, se debe guardar la identidad de las personas que han ofrecido sus huellas para colaborar con el proyecto.

2.3.2.1.2 Formato de las imágenes

Para la realización del proyecto se han empleado imágenes de impresiones dactilares tomadas a alumnos de tercer y cuarto curso del grado en Criminalística: Ciencias y Tecnologías Forenses de la Universidad de Alcalá de Henares.

Dichas impresiones fueron tomadas impregnando los dedos en grafito y plasmándolos en un papel de acetato. Para poder tratar las huellas tomadas, fueron escaneadas y pasadas a formato digital [Figura 4].

Figura 4. Fuente propia

Un aspecto importante a la hora de tratar las imágenes que serán clasificadas mediante redes convolucionales es el formato de archivo utilizado. Al digitalizar las imágenes, se debe decidir si serán comprimidas o no, por lo que se debe tener en cuenta la pérdida de información que esto puede ocasionar.

A. *Opción 1. Formato TIFF.*

Una de las posibles opciones de compresión que se ha contemplado es el formato TIFF (Tagged Imaged File Format) si se decide no realizar una gran compresión y por lo tanto no sufrir excesiva pérdida de información. Gracias a este formato que funciona mediante etiquetas y almacena mapas de bits, se consigue una gran calidad de las imágenes; sin embargo, uno de los problemas que se plantean es el espacio que ocupan [17].

Por lo tanto, una de las ventajas que supone guardar las imágenes de huellas dactilares con formato TIFF es que no se perderá la calidad necesaria para apreciar cada una de las crestas capilares de la huella, así como cualquier patrón o punto característico que pueda ayudar a la clasificación.

B. Opción 2. Formato JPEG.

Por otro lado, se ha planteado optar por un formato en el que se pierda calidad de imagen, pero que no ocupen excesivo espacio al guardarlas. Esta opción podría ser interesante para aquellos casos en los que se disponga de una base de datos de elevado tamaño, donde se buscará poder almacenar todas las imágenes en el menor espacio posible.

Un ejemplo de esta opción sería el formato JPEG (Joint Photographic Experts Group), que realiza la reducción de las imágenes eliminando aquella información que considere redundante al comparar los píxeles más cercanos. Esta compresión traducida a la calidad de las imágenes puede verse afectada, perdiéndose determinados detalles que podrían ser importantes [18].

C. Opción 3. Formato JPEG 2000.

A raíz de JPEG, se ha planteado por último el formato JPEG 2000, con la idea de mejorar la calidad de compresión. Esto se consigue, a diferencia del formato JPEG, porque esta compresión no produce pérdidas [19].

El formato JPEG 2000 emplea para ello lo que se conoce como Transformada Wavelet. Esta transformada permite representar funciones reteniendo tanto la escala como la información espacial [20].

¿Cómo funciona la transformada Wavelet?

Esta transformada no solo puede utilizarse para la compresión de imágenes, sino que también permite el filtrado para la eliminación de datos y presentar la información de manera breve. La Transformada Wavelet Discreta permite transformar un vector de datos que tengan determinada longitud, a otro vector de coeficientes wavelets de otra longitud, empleando precisamente las funciones wavelet, que realizan el producto escalar del vector de datos por cada una de las funciones base [20].

La fórmula de la DWT se define como:

$$W_f(s,\tau) = \int f\,(t)\Psi^*_{(s,\tau)}\,(t)dt$$

La función f(t) será descompuesta en las funciones base mencionadas anteriormente, las cuales responden a la siguiente fórmula [21]:

$$\Psi_{s,\tau}\,(t) = \frac{1}{\sqrt{s}}\,\Psi(\frac{t-\tau}{s})$$

- **s**: factor de escala
- $\boldsymbol{\tau}$: factor de traslación

La compresión de imágenes mediante la transformada Wavelet permite por lo tanto guardar imágenes descomponiéndolas en diferentes niveles y tipos de detalles [22], consiguiendo así una mejor calidad de las imágenes, pero sin ocupar mucho espacio, problema que supone el formato JPEG, pero que se solventa con el uso del formato JPEG 2000 [Figura 5].

Figura 5. Comparativa imagen JPEG frente a JPEG 2000 [23].

Concretamente, el procedimiento que realiza este formato al comprimir es el siguiente:

- Se realizan N niveles de descomposición en base a detalles aproximados, horizontales, verticales y diagonales de la imagen.
- Cuantificación de la imagen descompuesta, para maximizar los detalles necesarios y eliminar aquellos que no se necesiten.
- Para obtener la imagen transformada, se reconstruye mediante la transformada Wavelet inversa. Una vez se entiende el funcionamiento de los diferentes formatos de compresión, se debe escoger el que más se adapte a las necesidades del dataset.

Ejemplo de uso de Transformada Wavelet para digitalización de huellas.

Esta forma de compresión pasando de formato papel a formato digital para el almacenamiento de impresiones dactilares, es un procedimiento realizado por el FBI en los años 90. Se disponía de una gran cantidad de huellas tomadas a papel, más de 200 millones, por lo que era conveniente su digitalización.

Las imágenes estaban tomadas en escala de grises, por lo que su compresión suponía elevadas pérdidas. De esta forma, el FBI utilizó un tipo de compresión mediante transformadas Wavelet, que como se mencionaba anteriormente, implica un menor ratio de pérdidas [24].

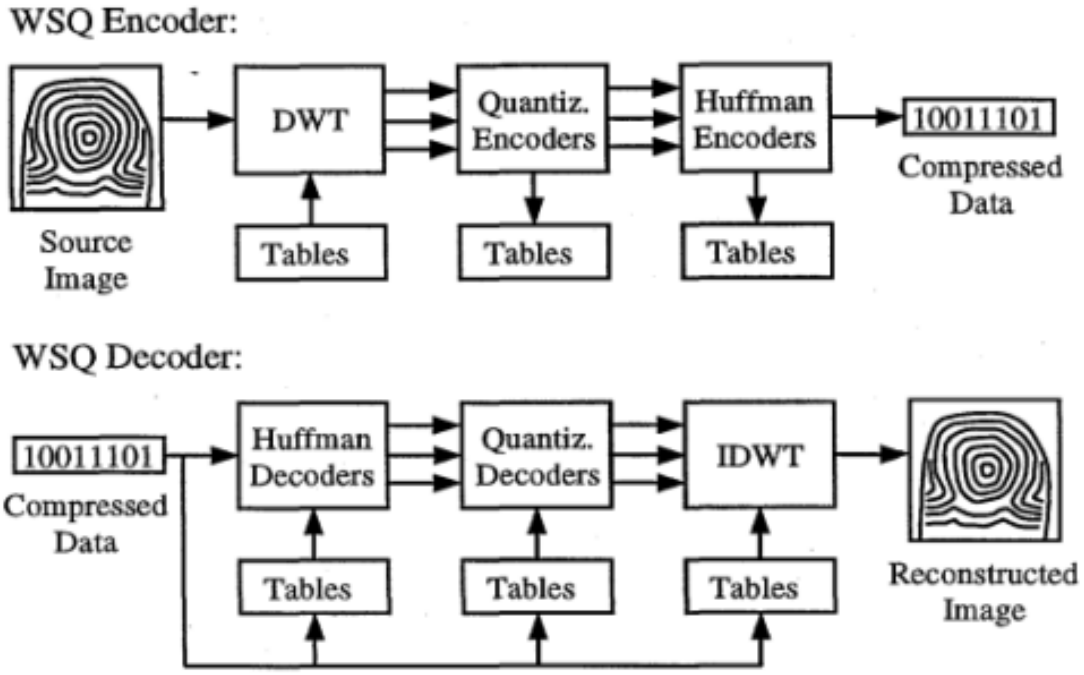


Figura 6. The FBI compression standard for digitized fingerprint images; esquema del algoritmo WSQ.

En la Figura 6 puede verse un esquema del algoritmo WSQ empleado por el FBI para la digitalización de huellas digitales. Consiste en tres fases para el codificador:

1. Descomposición por Transformada Wavelet discreta.
2. Cuantificación escalar, es decir, se selecciona un valor finito escalar para representar la muestra.
3. Codificación de entropía Huffman.

Formato de imagen escogido.

Con toda esta información, se podría concluir que el mejor formato con el que trabajar es el formato JPEG2000, ya que permite esa reducción del espacio a la hora de almacenar las imágenes pero sin comprometer la calidad de las mismas.

Sin embargo, las imágenes de las que se disponían fueron guardadas en un principio en formato JPG, por lo que, al haber usado en un primer momento un formato que supone pérdida de la información, no puede convertirse en un formato sin apenas pérdida, ya que no se recupera la información previamente eliminada.

Es por esto por lo que finalmente el formato elegido para tratar las imágenes es el formato JPG. Aún así, la calidad y los detalles perdidos no son muy notables.

2.3.2.1.3 Espacio de color del dataset

Inicialmente, las imágenes escaneadas contaban con espacio de color BN (blanco y negro), pero fueron transformadas a RGB, ya que para visualización (y no impresión) es un espacio de color más completo, cubriendo un espectro más amplio de colores [25].

Relacionemos el espacio de color elegido para las imágenes con la compresión que sufren las imágenes RGB, (con pérdida o sin pérdida, según el formato elegido).

Aplicar una compresión a una imagen RGB puede suponer cambios en los colores de dichas imágenes, modificándose su intensidad, tono y saturación [20].

Ahora bien, para que la imagen no se vea afectada de manera agresiva, es importante elegir el número óptimo de niveles de descomposición (Jmax) que se va a realizar para la compresión de las imágenes. Viene dado por la siguiente fórmula [26]:

$$J_{max} = fix(\log_2[\left(\frac{N}{N_w}\right) - 1])$$

- **N**: longitud de la señal
- **Nw**: longitud del filtro de descomposición
- **fix**: redondea el valor obtenido al entero más cercano

De esta forma, se podría conseguir la mejor calidad escogiendo el formato y el espacio de color más adecuados, sin comprometer detalles relevantes para la clasificación.

2.3.2.1.4 Resolución de las imágenes

Por último, cada imagen del dataset contiene unas dimensiones distintas. Al convertir las imágenes a RGB, todas comparten una tercera dimensión, la profundidad, que será de valor 3. Sin embargo, el tamaño de píxeles relativo a las dos primeras dimensiones debe ser el mismo.

Algunas de las imágenes de los alumnos de Criminalística han sido tomadas mediante un sensor de huellas dactilares; este sensor captura imágenes de dimensiones 256x288, por lo que serán las dimensiones elegidas para todas las imágenes del dataset inicial con las que será entrenado el modelo. El cambio de espacio de color y de dimensiones será realizado de forma simultánea para todas las imágenes del dataset mediante un script de Python.

2.3.2.2 Diseño del modelo

2.3.2.2.1 Librerías

Previo a la creación del algoritmo, se han definido las librerías empleadas para la elaboración del modelo. Las librerías usadas son las siguientes:

- **Módulo 'os'**. Este módulo nos permite acceder a los archivos del ordenador y poder trabajar con las imágenes de la base de datos, a través de la función 'listdir'. Además de esto, dicho módulo también permite crear carpetas o mostrar el directorio y las dimensiones en bytes de un archivo.
- **Módulo 'numpy'**. Proporciona estructuras de datos que permiten operar con matrices de forma más rápida, más eficiente y ocupando menos espacio de memoria, lo que supone una ventaja frente a las listas normales de Python [27].
- **Módulo 'matplotlib'**. Gracias a esta librería se pueden visualizar imágenes con la función 'image', y gráficas con la función 'pyplot' [28].
- **Módulo 'keras'**. Esta última librería es quizás la más importante y de la que más recursos se van a utilizar. Keras es un módulo que permite facilitar la creación de redes neuronales, y no lo hace de forma independiente, sino que funciona como una interfaz de programación de aplicaciones (API), que permite acceder a otras librerías que deben ser compatibles; en este caso, la librería compatible usada es TensorFlow [29].
- **Módulo Scikit-Learn**. Es una librería de Python exclusivamente para aprendizaje automático, que permite hacer un uso rápido y sencillo de diferentes algoritmos, funciones y métricas para la elaboración de modelos y redes neuronales.

2.3.2.2.2 Carga y tratamiento de los datos

Además de definir las librerías, también se han cargado los datos. Para ello, se han creado dos listas vacías: una para guardar las imágenes (Lista "images") y otra para guardar el tipo al que pertenece cada imagen (Lista "types"), es decir, el target, que será el valor a predecir.

A continuación, utilizando la función '*listdir*' de la librería 'os', se han recorrido todas las carpetas (Adeltos, Bideltos, Dextrodeltos y Sinistrodeltos) y se han guardado todas las imágenes en *images*, así como los tipos en *types* [Figura 12].

```
path = './BBDD HUELLAS/'

for tipo in os.listdir(path):
    for file in os.listdir(path + tipo):
        path_image = path + tipo + '/' + file
        images.append(image.imread(path_image))
        types.append(tipos_huella[tipo])
```

Figura 12.

Una vez cargadas las imágenes, mediante la función *train_test_split* de Scikit-Learn se ha divido el dataset:

- Un 80% de las imágenes ha sido destinado para el entrenamiento.
- El 20% restante ha sido destinado para realizar predicciones.

Por último, antes de comenzar con el modelo de entrenamiento, se ha realizado un último tratamiento al conjunto de imágenes:

- Conversión de las listas *train* y *test* iniciales en arrays de *numpy*, ya que resultan más fáciles de manipular y trabajar [Figura 13 A].
- Normalización de los datos, esto es, que los píxeles de las imágenes con las que se va a trabajar tengan valores comprendidos entre 0 y 1, por lo que ha sido suficiente con dividir entre 255 [Figura 13 B].
- Por último se definen las posibles clases que tendrá el target, en este caso 4 [Figura 13 C].

[A]

```
X_train = numpy.array(X_train)
X_test = numpy.array(X_test)
y_train = numpy.array(y_train)
y_test = numpy.array(y_test)
```

[B]

```
X_train = X_train.astype('float32')
X_test = X_test.astype('float32')
X_train /= 255
X_test /= 255
```

[C]

```
y_train = np_utils.to_categorical(y_train, 4)
y_test = np_utils.to_categorical(y_test, 4)
```

Figura 13

2.3.2.2.3 Modelo de red neuronal convolucional

Como se menciona en apartados anteriores, el tipo de red neuronal elegido para la clasificación de imágenes son las redes convolucionales.

Una vez realizado todo el procesamiento de las imágenes, debe desarrollarse la mejor estructura de las capas para lograr la mejor clasificación. Las capas empleadas han sido las siguientes:

- ***Convolution 2D***: con este tipo de capa se detectan patrones, extrayendo así una serie de características aplicando un filtro a la imagen. Debe incluir el número de neuronas, así como las dimensiones del filtro (ancho y alto) [30].
- ***MaxPooling***: se encarga de agrupar características de la imagen reduciendo así la resolución, y permitiendo después aplicar otra capa de convolución. Debe incluir los mismos valores que *Convolution2D* [30].
- ***Dropout***: utilizando esta capa, algunas de las neuronas son desactivadas de forma aleatoria. De esta forma, dichas neuronas no forman parte del entrenamiento ya que no pasan información a las siguientes neuronas, evitando así la dependencia de aprendizaje. Cuanto mayor sea su valor, mayor número de neuronas son desactivadas [31].
- ***Flatten***: se encarga de aplanar la matriz de entrada, pasando, por ejemplo, de dos dimensiones que puede tener un dato de entrada, a una sola dimensión [32].
- ***Dense***: sirve como conexión con la capa de salida, y requiere de una capa *Flatten* antes de aplicarse [33].

Se debe tener en cuenta que, tras cada conjunto de capas de convolución, se debe incorporar una capa de *MaxPooling*; además, las capas ocultas deben hilarse con la capa de salida mediante un filtro *Flatten*. Se han combinado todas estas capas para conseguir los mejores modelos que consigan buenos resultados de clasificación.

Además de las capas que conforman el algoritmo, se han estudiado y seleccionado otros parámetros que deben tenerse en cuenta a la hora de compilar el modelo:

- **Función de activación**. La mejor opción que se ha empleado es la función *ReLu*, la más apropiada para trabajar con redes neuronales convolucionales. Esto supone la eliminación de todos los valores negativos, lo que proporciona un trabajo y unos resultados óptimos. Se ha utilizado la misma función en todo el modelo, a excepción de la última capa, que utiliza la función *softmax*, función principalmente para la capa de salida en casos de clasificación múltiple [34].
- ***Padding***. Como se menciona anteriormente, las capas poseen unas dimensiones a las que se pueden asignar los valores deseados. Estas capas se van a ir aplicando por toda la imagen, recorriendo todos los píxeles, por lo que, en los extremos habrá parte del filtro que no englobe ninguna parte de la imagen. Con el '*padding'* se completa con ceros esa

parte ‘vacía’ del filtro para evitar pérdida de información y conseguir que la matriz de salida no se reduzca de tamaño [35].

- **Función de pérdida**. Es importante seleccionar el tipo de pérdida; *categorical_crossentropy* ha sido la función elegida, pues es la más común en los modelos de IA destinados a la clasificación de objetos. Muestra la suma de los errores de predicción cometidos en cada epoch [36].
- **Optimizador**. En cuanto al optimizador, el más usado en CNN es el optimizador Adam, que actualiza los pesos de las neuronas a medida que se va realizando el entrenamiento, que debe incluir un valor para el ‘learning ratio’ [37].
- **Métricas**. Por último, se ha definido la métrica a emplear. Por defecto, para los algoritmos de clasificación múltiple suele utilizarse la métrica *accuracy*; sin embargo, en este caso el dataset utilizado es desbalanceado, esto es, hay clases con un mayor número de imágenes que otras. Por eso, para esta casuística es recomendado utilizar otra métrica, como la *f1_score*, *precision* y *recall*.

Por último, se han determinado los parámetros finales que permiten realizar el entrenamiento. Estos parámetros son los siguientes:

- ***Epochs***. Es el número de veces que se quiere aplicar el modelo a las imágenes, por lo que se ha ido variando su valor para encontrar el más óptimo [38].
- ***Batch size***.Este hiperparámetro permite definir el número de imágenes con el que se va a ir trabajando antes de actualizar los parámetros del modelo [38]
- ***Verbose***. Con esta opción, se puede obtener información (porcentaje de pérdida y precisión) después de ejecutarse cada ‘epoch’, para lo que se ha elegido el valor 1; si se introduce el valor 0, no se muestra dicha información [39].

Además de la fase de entrenamiento, se ha realizado una fase de evaluación, empleando la función *evaluate* de Keras. Para ello, se han utilizado los conjuntos de *test* creados anteriormente; se obtiene con esto una puntuación para la función de pérdida, *accuracy, f1_score*, *precision* y *recall*.

2.4 Resultados

Con todos los parámetros mencionados en el apartado anterior, se han elaborado un total de 14 modelos, con el fin de obtener aquel que mejor se ajuste a la clasificación deseada. Estos modelos se han diseñado para entrenar con el dataset formado por las 1260 imágenes provenientes de las dos fuentes mencionadas en el apartado 2.3.1.

Los primeros 8 modelos creados han sido más sencillos, contando simplemente con un par de capas de convolución, una capa de *pooling* y un par de capas densas. Para estos sencillos modelos se han ido modificando los valores relativos al número de filtros y al tamaño del filtro.

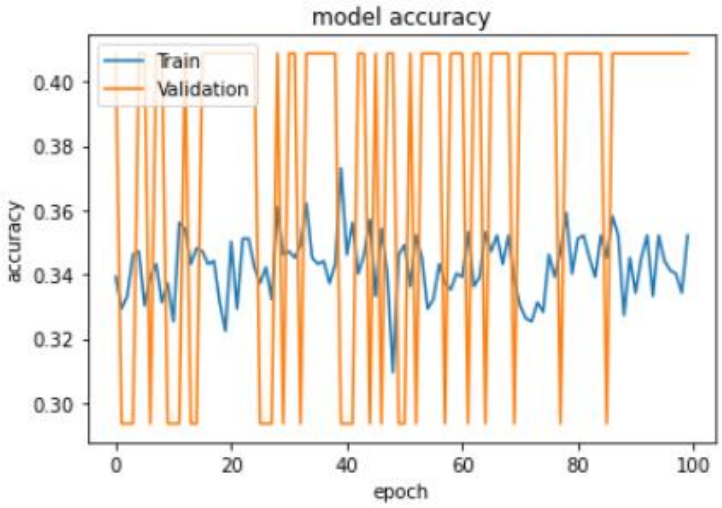
Debido al gran tamaño de las imágenes (256x288), se han empleado tamaños de filtros grandes, como de 7x7 y 9x9.

Los resultados obtenidos para dichos modelos se pueden observar en la siguiente tabla:

	Loss	Accuracy	F1 Score	Precision	Recall
Modelo 1	1.3024	0.3769	0.0	0.0	0.0
Modelo 2	12.879	0.3055	0.3046	0.3046	0.3046
Modelo 3	1.2676	0.4087	0.0	0.0	0.0
Modelo 4	1.3024	0.3769	0.0	0.0	0.0
Modelo 5	1.4953	0.3650	0.0212	0.1145	0.0117
Modelo 6	1.2320	0.4166	0.0	0.0	0.0
Modelo 7	53.99	0.3333	0.3309	0.3309	0.3309
Modelo 8	1.4953	0.3650	0.0212	0.1145	0.0117

Observando los resultados, se puede concluir que los mejores resultados se dan para los modelos 3 [Figura 14] y 6 [Figura 15].

[A]



[B]

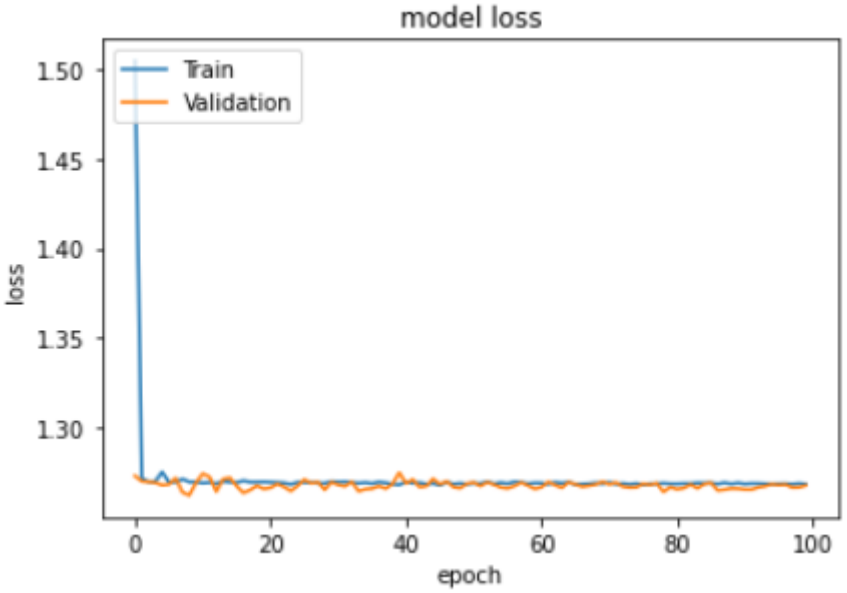
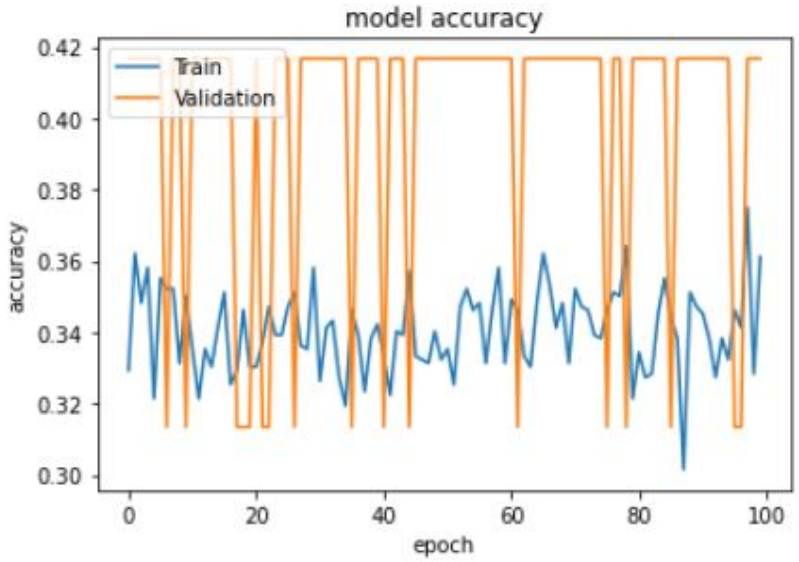


Figura 14. [A] Función de pérdida del modelo 3; [B] Accuracy del modelo 3

[A]



[B]

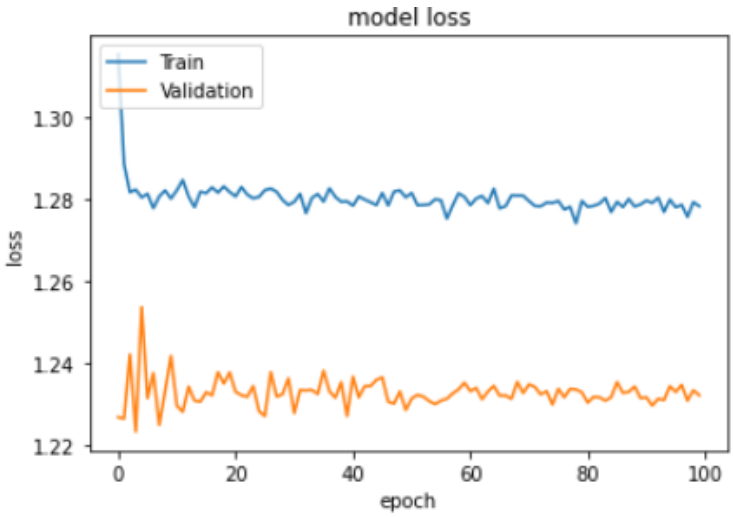


Figura 15. [A] Función de pérdida del modelo 6; [B] Accuracy del modelo 6

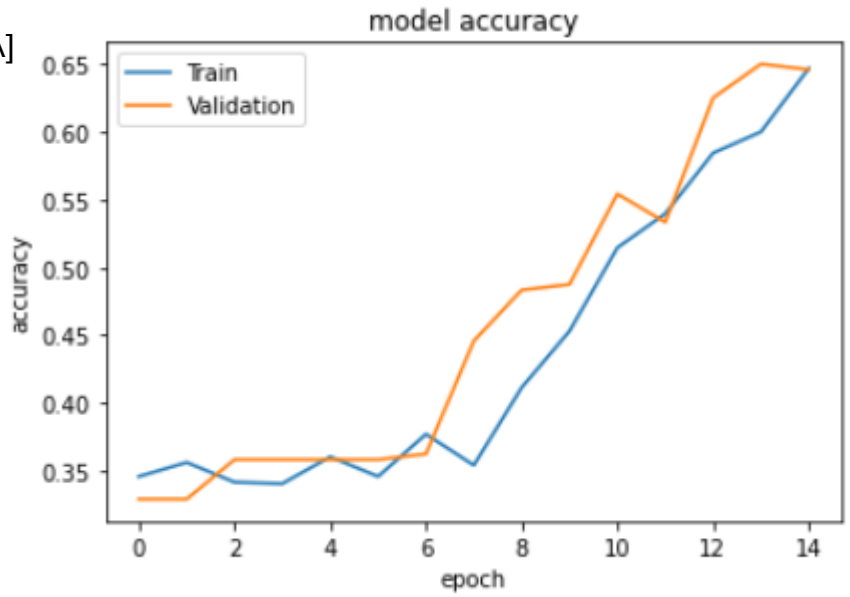
A pesar de obtener algunos modelos ligeramente mejores que los demás, los resultados no reflejan un buen entrenamiento, por lo que también se han elaborado modelos para entrenar con las 1200 imágenes obtenidas de la misma fuente, eliminando las 60 imágenes obtenidas de la base de datos del FVC2000, que tienen un formato diferente.

Los resultados de los modelos creados para este segundo conjunto de datos se ven reflejados en la siguiente tabla:

	Loss	Accuracy	F1 Score	Precision	Recall
Modelo 9	1.2788	0.3375	0.0	0.0	0.0
Modelo 10	1.2337	0.3916	0.0	0.0	0.0
Modelo 11	1.2914	0.3083	0.0	0.0	0.0
Modelo 12	0.9842	0.6458	0.6293	0.6754	0.5898

Si se observan los resultados, puede verse cómo el modelo 12 obtiene resultados considerablemente mejores respecto a todas las métricas. Esto se ha conseguido modificando el valor del *learning ratio* del optimizador Adam empleado.En las gráficas de la Figura 16 puede observarse cómo el modelo no consigue salir de un mínimo local para llegar al mínimo real, y cuando logra salir, el modelo mejora notablemente y la función de pérdida cae.

[A]



[B]

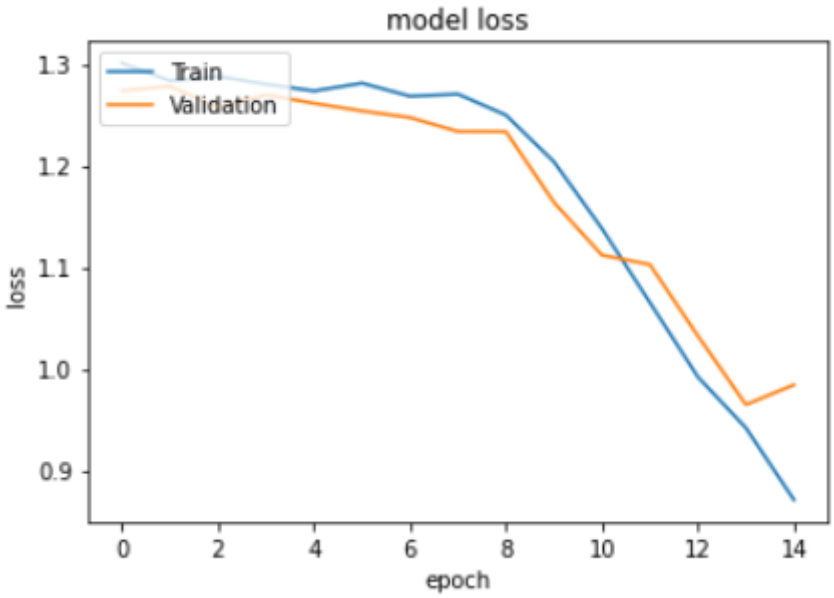


Figura 16. [A] Función de pérdida; [B] Accuracy

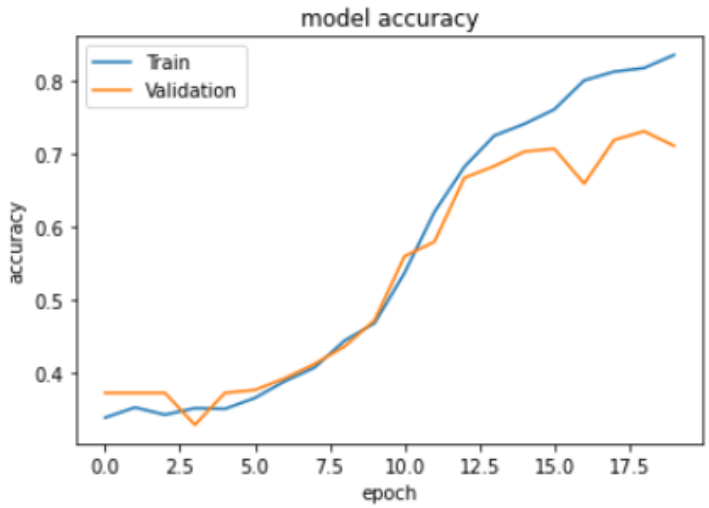
Debido a la obtención de estos valores, se plantea si el modelo no estaba entrenando de manera eficaz debido a este valor, y no al dataset utilizado. Es por eso que se vuelven a ejecutar los mejores modelos obtenidos anteriormente para el dataset completo, pero modificando dicho parámetro del optimizador.

Los resultados que se han obtenido se observan a continuación:

	Loss	Accuracy	F1 Score	Precision	Recall
Modelo 13	0.8857	0.7103	0.7188	0.7487	0.6919
Modelo 14	1.1154	0.7738	0.7799	0.7867	0.7734

Los resultados para estos modelos muestran un mejor entrenamiento debido al valor del learning ratio (0.00001). En la Figura 17 puede observarse de nuevo para el modelo 13 cómo la función de pérdida cae cuando consigue salir de un mínimo local.

[A]



[B]

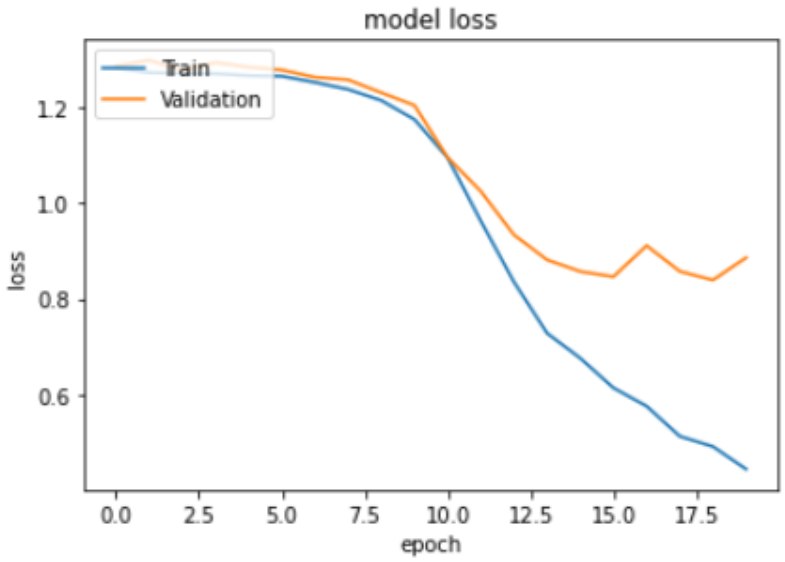
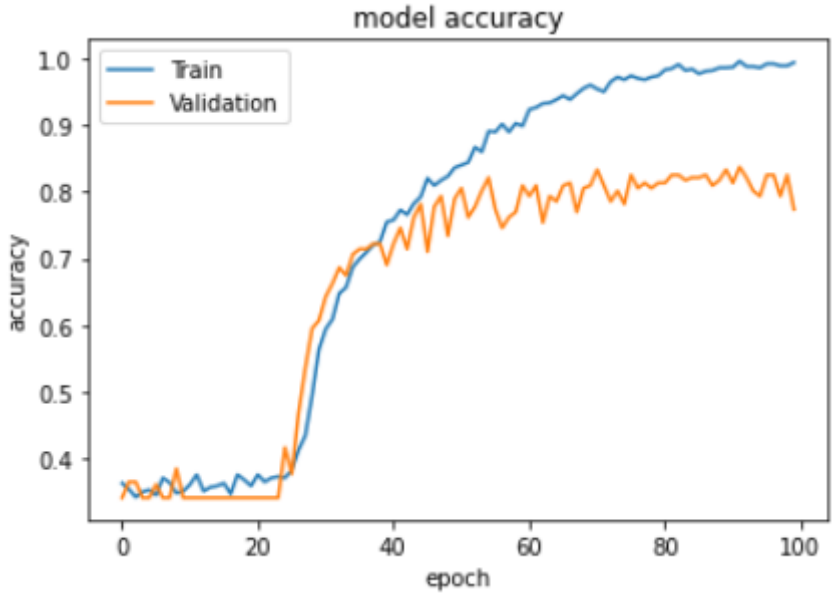


Figura 17. [A] Función de pérdida; [B] Accuracy

El modelo 14 también obtiene buenos resultados, pero en las gráficas de la Figura 18 puede observarse cómo comienza a sobreentrenar aproximadamente a partir de la epoch 50.

[A]



[B]

Figura 18. [A] Función de pérdida; [B] Accuracy

Con estos resultados, se han desarrollado una serie de conclusiones que se encuentran en el siguiente capítulo.

Capítulo 3. Conclusiones

3.1 Conclusiones del modelo seleccionado

Las conclusiones extraídas tras la realización del proyecto se han dividido en diferentes aspectos para comprender mejor los resultados obtenidos. Las conclusiones se detallan a continuación:

3.1.1 Conclusiones relativas al dataset

Si observamos las características del dataset empleado para el entrenamiento, se puede deducir lo siguiente:

- El tamaño del dataset es pequeño, ya que solo se ha contado con un total de 1260 imágenes para el entrenamiento de los modelos elaborados. Esto se ha debido a la dificultad que supone la adquisición de impresiones dactilares de gente voluntaria, al tratarse de un dato biométrico sensible y difícil de tomar debido a los métodos de obtención.
- El dataset empleado es considerablemente desbalanceado, ya que el número de imágenes para las clases Dextrodelto y Sinistrodelto es mucho mayor que para las clases Adelto y Bidelto. Esto sucede porque las dos primeras clases son más abundantes en la población, lo que supone una mayor dificultad encontrar una proporción similar de los cuatro tipos [Figura 19].
- Las imágenes proceden de dos fuentes diferentes. La mayoría de las imágenes fueron tomadas con grafito y papel de acetato, y posteriormente escaneadas y digitalizadas; el resto de las imágenes (un 5% del total) fueron obtenidas de una base de datos. Esto supone una diferencia de características de las huellas en cuanto al método de captura, por lo que el entrenamiento se complica, ya que el modelo debe aprender a clasificar dos tipos de imágenes completamente diferentes.

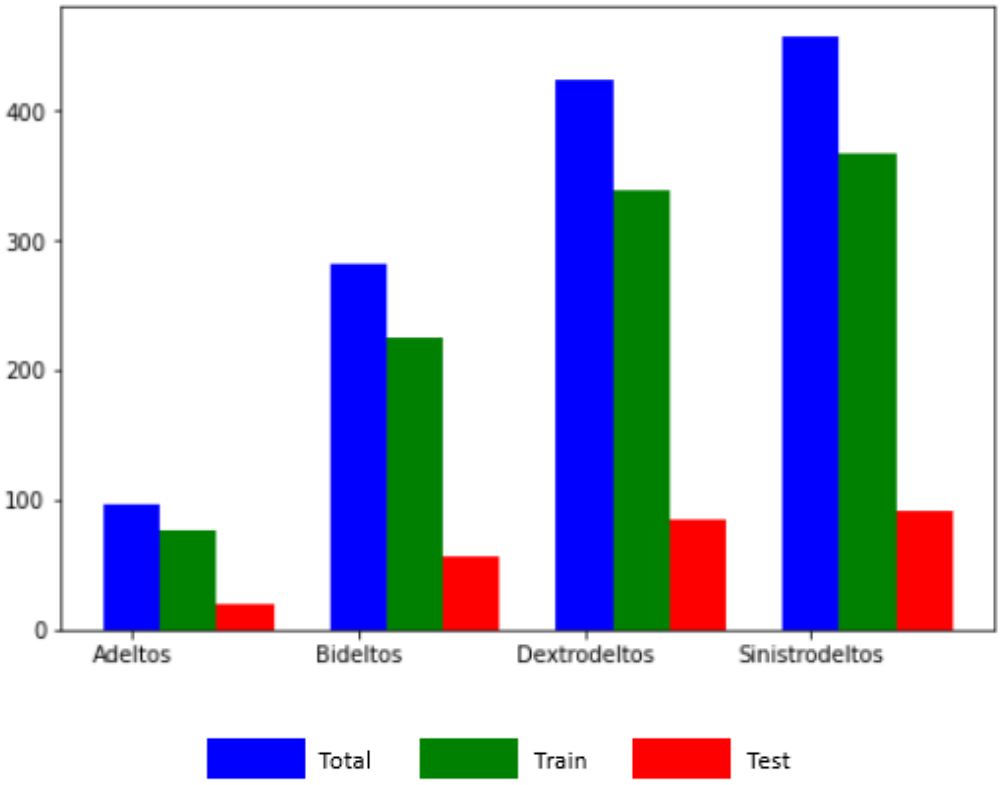


Figura 19

3.1.2 Conclusiones relativas a las métricas

Como se ha mencionado anteriormente, el dataset empleado para el entrenamiento es desbalanceado. Esto no solo supone un problema a la hora de realizar el entrenamiento del modelo, sino que también influye en la selección de las métricas a emplear.

Como cualquier problema de clasificación en múltiples clases, la métrica más óptima para medir el nivel de eficacia del modelo es la *accuracy*; sin embargo, para casos donde el dataset empleado esté desbalanceado, no resulta del todo útil. Esto sucede porque aunque no consiga clasificar de forma correcta las clases minoritarias, sí lo hará con las más abundantes, lo que supone un alto porcentaje de acierto, ya que la mayoría de las imágenes a clasificar corresponden con esas clases mayoritarias.

No obstante, y como se comentaba en las conclusiones relativas a la base de datos, se trata de un conjunto de imágenes reducido, y por lo tanto, los conjuntos de *train* y *test* son también pequeños. Es cierto que la métrica de *accuracy* no es recomendable para *datasets* desbalanceados; además de esto, si sumamos el problema de los conjuntos pequeños, cuando el modelo clasifica de forma errónea muy pocas imágenes, resultan una gran proporción respecto al total del conjunto, que es muy pequeño.

Por el contrario, cuando el modelo consigue clasificar bien sin demasiados errores, ambas métricas empleadas, la *accuracy* y la *f1 score*, obtienen buenos

resultados muy similares, lo que permite apoyarse en dos métricas para concluir una buena clasificación.

3.1.3 Conclusiones relativas a los parámetros

En cuanto a los parámetros empleados para la elaboración de los modelos, se han extraído conclusiones para dos de ellos:

- ***Learning ratio***. Como se ha podido observar en los resultados obtenidos, para los primeros modelos no se obtenían buenas métricas. Al observar las gráficas, se aprecia cómo el modelo en la fase de entrenamiento no consigue mejorar debido al estancamiento en un mínimo local, no consiguiendo llevar la función de pérdida al mínimo real. Modificando el valor del *learning ratio* se ha conseguido superar esta barrera; en la Figura 20 puede observarse cómo la elección de este valor es crucial para un buen entrenamiento, ya que un *learning ratio* demasiado bajo puede suponer un tiempo muy elevado para que el algoritmo converja; sin embargo, un valor demasiado alto puede llevar también a la dificultas de converger e impedir que alance el mínimo real.. Se concluye por tanto que su manipulación ha sido clave en este proyecto para la obtención de buenos resultados.

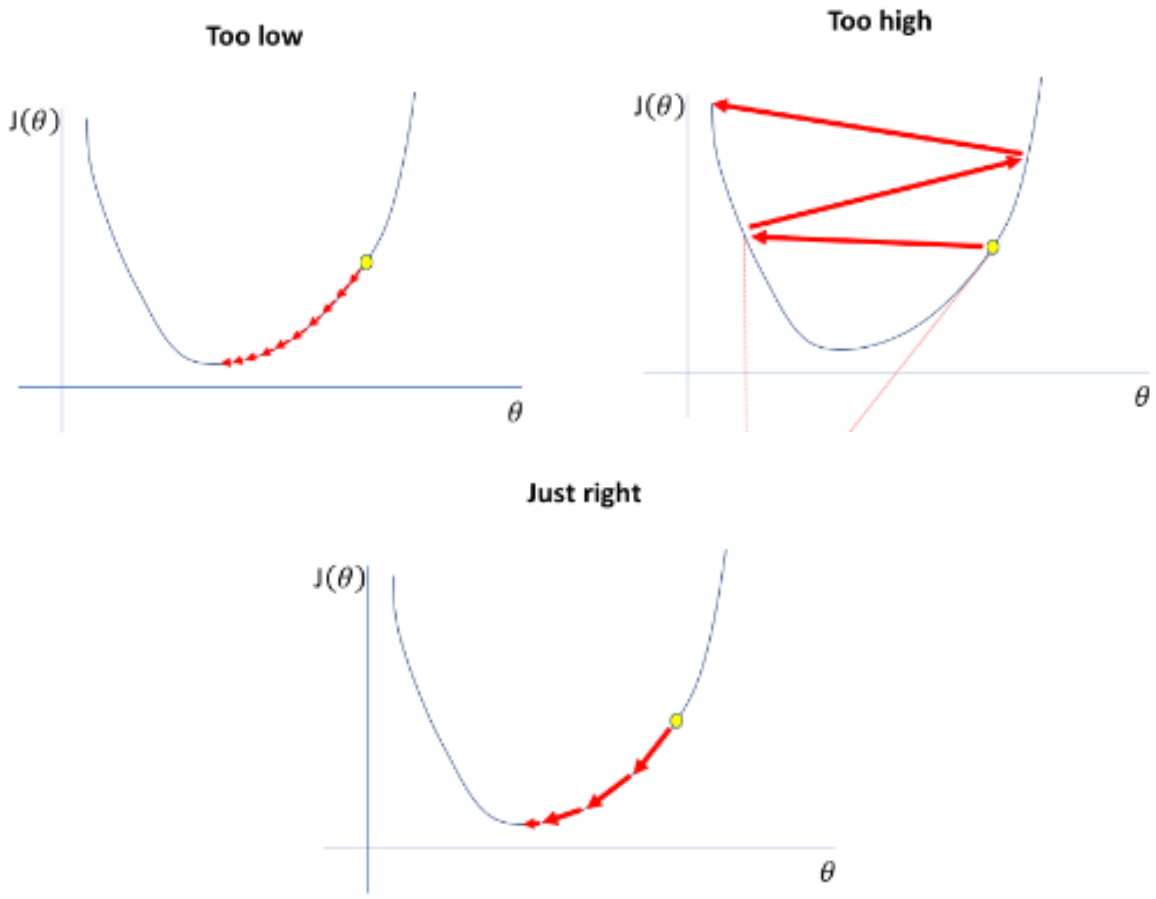


Figura 20 [40]

- ***Epochs***. Por último, el número de *epochs* también ha sido muy importante para la elaboración de unos buenos resultados.Puede observarse como por ejemplo, en el modelo 14 se obtiene una *accuracy* de 0.79, pero al ver la gráfica correspondiente [Figura 18B] se puede apreciar cómo se alcanzan valores mayores que dicho resultado, hasta llegar incluso a 0.98. Esto sucede porque el modelo está sobreentrenando, aproximadamente a partir de la *epoch* número 50. Por lo tanto, este parámetro se ha tenido en cuenta para evitar un sobreajuste.

3.2 Estudio de visualización sobre los resultados (Power BI)

Por último, con el fin de aportar una visión global sobre el desarrollo del proyecto y sus resultados, se han elaborado dos *dashboards* empleando Power BI.

El primer *dashboard* [Figura 21] muestra una visión sobre el dataset empleado, incluyendo una breve presentación de las diferentes clases de huellas que se han usado para el entrenamiento mediante un mapa de árbol, con una tarjeta interactiva que muestra la cantidad exacta de imágenes de cada tipo. Además, se muestra un gráfico de barras agrupadas que permite visualizar una aproximación de distribución del dataset en los conjuntos de *train* y *test*.

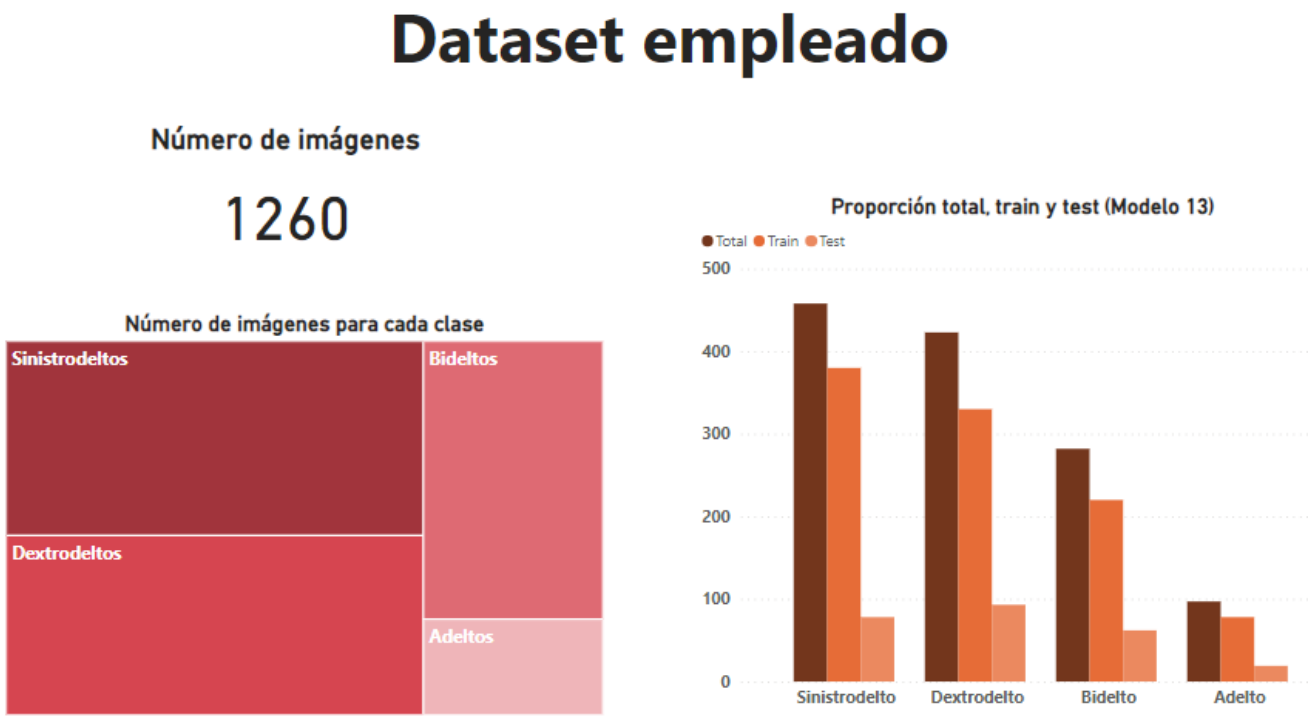


Figura 21

El segundo *dashboard* [Figura 22] refleja los resultados obtenidos tras el entrenamiento de los diversos modelos. Se han escogido los cuatro mejores modelos: un primer gráfico de barras muestra la función de pérdida para esos cuatro modelos, visualizándose qué modelos sobresalen respecto a otros; por otro lado, un segundo gráfico de barras muestra la misma comparativa, pero

relativa a los valores de *accuracy* obtenidos. De esta forma, puede concluirse que el mejor modelo es el 13, ya que simplemente hay una diferencia del 2% con respecto al modelo 14, pero presenta menor función de pérdida.

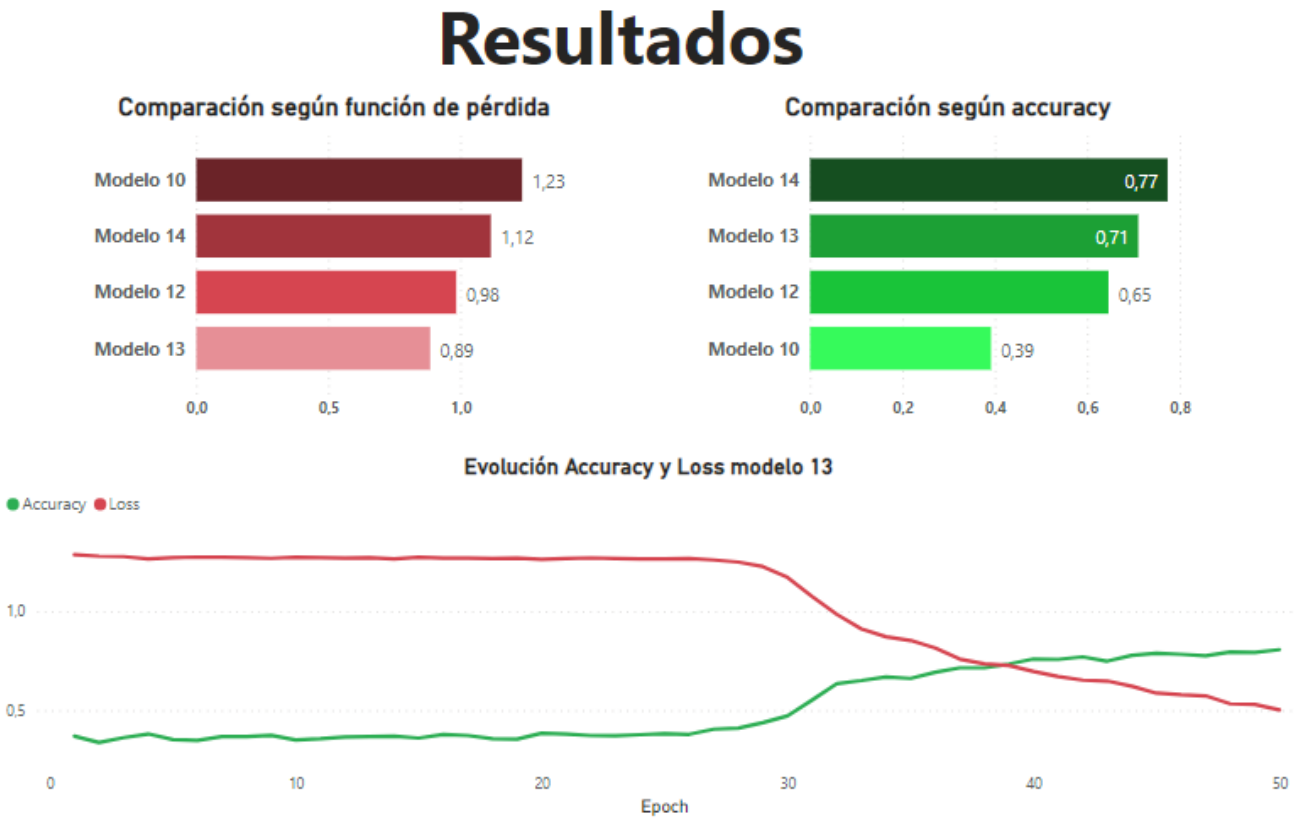


Figura 22

Gracias a estas conclusiones visuales, se permite una mayor comprensión del proyecto.

BIBLIOGRAFÍA

[1] INFANTE TOPA, Milena. La importancia de la dactiloscopia en la investigación criminal. Proyecto de investigación de doctorado, Universidad la Gran Colombia, 2015.

[2] Lofoscopia. Directorio forense [en línea]. [sin fecha]. Disponible en: https://directorioforense.com/index.php/lofoscopia/

[3] Código Taiho. Cofactor [en línea]. [sin fecha]. Disponible en: https://cofactor.io/es/m/081t4t

[4] Juan Vucetich. Biografías [en línea]. [sin fecha]. Disponible en: https://www.biografias.es/famosos/juan-vucetich.html

[5] HUTCHINS, Laura A. Sistemas de clasificación de crestas de fricción. En: El libro de referencia de las huellas dactilares. Washington, 2002.

[6] RODRÍGUEZ ARRIETA, Aldo Neyl. Perfil dermatoglífico y condición física de jugadores adolescentes de futbol. Universidad Nacional de La Plata. 2017, 19(2). ISSN 2314-2561.

[7] JUÁREZ MEZA, José Manuel. Dactiloscopia. Criminalística México [en línea]. [sin fecha]. Disponible en: https://criminalistica.mx/descargas/documentos/pdf/DactiloscopiaJMJM.pdf

[8] ROBLEDO ACINAS, María del Mar, José Antonio SÁNCHEZ SÁNCHEZ y Raquel AGUILAR UNGIL. Estudio de las frecuencias de los tipos dactilares y de los puntos característicos en dactilogramas de población española. Derecho y cambio social. 2012. ISSN 2224-4131.

[9] Puntos característicos de las huellas digitales. El legado en la escena del delito [en línea]. 7 de mayo de 2014. Disponible en: http://ellegadoenlascenadeldelito.blogspot.com/2014/05/puntos-caracteristicos-de-las-huellas.html

[10] ABRISHAMBAF, Reza y Hasan DEMIREL. A Fully CNN Based Fingerprint Recognition System. 11th International Workshop on Cellular Neural Networks and their Applications. 2018, 14–16.

[11] Deep Learning: clasificando imágenes con redes neuronales. Lis Data Solutions [en línea]. [sin fecha]. Disponible en: https://www.lisdatasolutions.com/es/blog/deep-learning-clasificando-imagenes-con-redes-neuronales/

[12] Intro a las redes neuronales convolucionales. Bootcamp AI [en línea]. 23 de noviembre de 2019. Disponible en: https://bootcampai.medium.com/redes-neuronales-convolucionales-5e0ce960caf8

[13] ¿Cómo funcionan las Convolutional Neural Networks? Visión por Ordenador. Aprende Machine Learning [en línea]. 29 de noviembre de 2018. Disponible en: https://www.aprendemachinelearning.com/como-funcionan-las-convolutional-neural-networks-vision-por-ordenador/

[14] BARRIOS, Juan. Redes Neuronales Convolucionales. Health Big Data [en línea]. [sin fecha]. Disponible en: https://www.juanbarrios.com/redes-neurales-convolucionales/

[15] MALTONI, D. Fingerprint Verification Competition [Dataset de imágenes]. 2009. Londres.

[16] Datos biométricos: qué son y su tratamiento en la protección de datos (RGPD y LOPDGDD). Grupo Ático 34 [en línea]. [sin fecha]. Disponible en: https://protecciondatos-lopd.com/empresas/datos-biometricos-rgpd/#Huella_dactilar

[17] LÓPEZ JURADO, Carlos. Características del formato TIFF. CCM [en línea]. 29 de enero de 2021. Disponible en: https://es.ccm.net/contents/724-el-formato-tif

[18] JACKSON, Brian. JPG vs JPEG: Entendiendo el formato de archivo de imagen más común. Kinsta [en línea]. 18 de junio de 2021. Disponible en: https://kinsta.com/es/blog/jpg-vs-jpeg/

[19] JPEG vs. JPEG 2000. Adobe [en línea]. [sin fecha]. Disponible en: https://www.adobe.com/es/creativecloud/file-types/image/comparison/jpeg-vs-jpeg-2000.html

[20] AZOR MONTOYA, J. R. La transformada Wavelet. Revista de la Universidad de Mendoza, 2013.

[21] Introducción a la Transformada Wavelet. Universidad Nacional del Centro de la provincia de Buenos Aires, 2006.

[22] SONG, M. S. Wavelet Image Compression. Contemporaty Mathematics.

[23] Optimizando imágenes con nuevos formatos: WebP y JPEG 2000. Fotografía eCommerce [en línea]. 30 de abril de 2019. Disponible en: https://www.fotografiaecommerce.com/blog/optimizando-imagenes-formatos-webp-jpeg2000/

[24] BRISLAWN, Christopher M. y Jonathan N. BRADLEY. The FBI compression standard for digitized fingerprint images. Los Alamos National Laboratory. 1996.

[25] SARA. Diferencia entre RGB y CMYK: ¿Cómo y cuándo elegir? Stampaprint [en línea]. 10 de marzo de 2016. Disponible en: https://www.stampaprint.net/es/blog/acerca-de-la-impresion/diferencia-entre-rgb-y-cmyk-como-y-cuando-

elegir#:~:text=Para%20resumir,%20si%20un%20archivo,espectro%20de%20colores%20más%20grande.

[26] KEMARI, Yusseff. How to choose wavelet level decomposition? Research Gate [en línea]. 2 de noviembre de 2016. Disponible en: https://www.researchgate.net/post/How-to-choose-wavelet-level-decomposition

[27] Introducción a Numpy. Aprende IA [en línea]. 2021. Disponible en: https://aprendeia.com/introduccion-a-numpy-python-1/

[28] Matplotlib: Visualization with Python. Matplotlib [en línea]. 2012. Disponible en: https://matplotlib.org/

[29] Keras: biblioteca de código abierto para crear redes neuronales. Ionos [en línea]. 8 de octubre de 2020. Disponible en: https://www.ionos.es/digitalguide/online-marketing/marketing-para-motores-de-busqueda/que-es-keras/

[30] Keras Tutorial: The Ultimate Beginner's Guide to Deep Learning in Python. Elite Data Science [en línea]. 7 de julio de 2022. Disponible en: https://elitedatascience.com/keras-tutorial-deep-learning-in-python

[31] RODRÍGUEZ, Vicente. Dropout y Batch Normalization. Vicente Rodríguez Blog [en línea]. 9 de noviembre de 2018. Disponible en: https://vincentblog.xyz/posts/dropout-y-batch-normalization

[32] Keras - Flatten Layers. Tutorials Point [en línea]. [sin fecha]. Disponible en: https://www.tutorialspoint.com/keras/keras_flatten_layers.htm

[33] The Keras Dense Layer. Sparrow Computing [en línea]. 31 de diciembre de 2020. Disponible en: https://sparrow.dev/keras-dense-layer/

[34] CALVO, Diego. Función de activación – Redes neuronales. Diego Calvo [en línea]. 7 de diciembre de 2018. Disponible en: https://www.diegocalvo.es/funcion-de-activacion-redes-neuronales/

[35] What is padding in neural network? Geeks for Geeks [en línea]. 18 de marzo de 2022. Disponible en: https://www.geeksforgeeks.org/what-is-padding-in-neural-network/

[36] KUMAR, Ajitesh. Keras – Categorical Cross Entropy Loss Function. Data Analytics [en línea]. 28 de octubre de 2020. Disponible en: https://vitalflux.com/keras-categorical-cross-entropy-loss-function/

[37] BILOGUR, Aleksey. Keras optimizers. *Kaggle* [en línea]. 2018. Disponible en: https://www.kaggle.com/code/residentmario/keras-optimizers/notebook

[38] BROWNLEE, Jason. Difference Between a Batch and an Epoch in a Neural Network. Machine Learning Mastery [en línea]. 10 de agosto de 2022. Disponible

en: https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/#:~:text=The%20batch%20size%20is%20a%20number%20of%20samples%20processed%20before,samples%20in%20the%20training%20dataset.

[39] Model training APIs. Keras [en línea]. [sin fecha]. Disponible en: https://keras.io/api/models/model_training_apis/

[40] Summary: True Gradient Descent. AI Summary [en línea]. 25 de marzo de 2021. Disponible en: https://www.ai-summary.com/summary-true-gradient-descent/