UNIVERSIDAD SEÑOR DE SIPÁN

FACULTAD DE INGENIERÍA, ARQUITECTURA Y URBANISMO

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

TESIS

DESARROLLO DE UN MÉTODO DE IDENTIFICACIÓN DE PERSONAS MEDIANTE EL PROCESAMIENTO DE IMÁGENES DIGITALES DE LA IMPRESIÓN PALMARIAS

PARA OPTAR EL TITULO PROFESIONAL DE INGENIERO DE SISTEMAS

Autor:
Bach. Céspedes Ordóñez Barny Napoleón
ORCID: https://orcid.org/0000-0003-3566-2736

Asesor:
Mg. Mejia Cabrera Heber Ivan
ORCID: https://orcid.org/ 0000-0002-0007-0928

Línea de Investigación:
Infraestructura Tecnología, y Medio Ambiente.

Pimentel – Perú
2022

DESARROLLO DE UN MÉTODO DE IDENTIFICACIÓN DE PERSONAS MEDIANTE EL PROCESAMIENTO DE IMÁGENES DIGITALES DE LA IMPRESIÓN PALMARIAS

APROBACIÓN DEL JURADO

Presentado por:

_____

Bach. Céspedes Ordóñez Barny Napoleón
Autor

Aprobado por:

_____

Mg. Mejia Cabrera Heber Ivan
Asesor

_____	_____
Mg. Ramos Moscol Mario Fernando	Mg. Mejia Cabrera Heber Ivan
PRESIDENTE	SECRETARIO

_____

Mg. Bravo Ruiz Jaime Arturo
VOCAL

Fecha de presentación: Febrero 2022

DEDICATORIAS

A mis padres, los seres más
hermosos que Dios Padre me regaló.

AGRADECIMIENTOS

. A mi asesor el Mg. Ivan Mejia por
entregar su amplio conocimiento en investigación. A
las personas que me apoyaron con su tiempo
de una manera amable y desinteresada
para realizar mi más anhelado sueño.

RESUMEN

Los métodos de identificación más comunes de reconocimiento de personas en la actualidad presentan problemas de detección, esto por las huellas dactilares que son borrosas por el trabajo, y a esto sumado que estos métodos están siendo vulnerados, hoy en día hacen necesario realizar nuevos métodos de identificación. El objetivo de esta investigación es: Desarrollar un método por medio de un análisis comparativo de los métodos de reconocimiento de las palmas de las manos con modelos producido por redes neuronales convolucionales, para resolver problemas de reconocimiento, la presente investigación se origina por el estudio realizado en las bases a la teoría científica, estado del arte, los problemas diversos existentes en la implementación de técnicas de reconocimiento de huellas palmarias, donde se encuentran métodos que son eficaces y estos pueden procesar los resultado rápidamente pero con problemas de identificación, por lo que, en este estudio se plantea desarrollar un análisis de comparación de seis métodos de reconocimiento de palmas de manos con procesamiento de imágenes, para conocer qué método tiene un mejor desempeño en la tarea de entrenamiento y reconocimiento de las huellas palmarias.

Para realizar el trabajo de esta investigación se utilizaron las fotografías digitales de palmas de manos de 100 personas que fueron registradas de manera individual, como muestra para el desarrollo de la prueba de las 6 redes neuronales convolucionales seleccionadas las cuales son: la VGG16, VG19, ResNet50, MobileNetV2, Xception y DenseNet121. Además, se creó un propio protocolo de bioseguridad por encontrarnos en tiempos de pandemia y lineamientos de registro de imágenes para luego grabarlas de tal manera que pudieran procesarse por 06 de los métodos escogidos. Como parte de la evaluación de resultados se implementó un script con python donde se mide los indicadores como: Tiempo de respuesta, Precisión, Exactitud, Recall, Valor F.

Al término de experimentar con 6 redes convoluciones se concluyó la red neuronal con mejores resultados para esta tarea fue la RestNet50 ya que su porcentaje de acuraccy fue de 99% y el más alto de los demás métodos.

PALABRAS CLAVE: Huella Palmaria, Redes Convolucionales, biometría, Python.

ABSTRACT

The most common biometric methods of recognition of people nowadays present problems of recognition this because the fingerprints are blurred and illegible by changes in aspects of the fingerprints by the workers of the day to day that the field personnel is exposed and ah this added that these methods are being violated nowadays make necessary to carry out new methods of identification. The main objective of the present research is to make a comparative analysis of the methods of recognition of the palms with convolutional neural networks to solve recognition problems in a controlled environment, this research arises from the study made on the basis of scientific theory, state of the art, the various problems existing in previous works with the implementation of methods of palm print recognition. Where methods are found that are effective and these can process the results quickly but with some problems of identification, therefore, this study proposes to develop a comparison analysis of six methods of palm recognition with image processing to know which method achieves a superior performance in the task of training and palm print recognition.

To accomplish the task of this research , we used the digital photographs of palms of 100 people that were acquired by ourselves, as a sample for the testing of the 6 selected methods which are the convolutional neural networks VGG16, VG19, ResNet50, MobileNetV2, Xception and DenseNet121. In addition, an own biosafety protocol was created because we are on time of pandemic and an image acquisition protocol to then store them in a way that could be processed by the selected methods.

For the evaluation of results a python script was implemented where indicators such as, Response time, Accuracy, Recall, F-value are measured.

After experimenting with 6 convolution networks, the neural network with the best results for this task was the RestNet50 since its acuraccy percentage was 99% and the highest of the other methods.

KEYWORDS: Palmprint, Convolutional Networks, biometrics, Python.

ÍNDICE

I. INTRODUCCIÓN

1.1. Realidad Problemática.

Actualmente existe un gran incremento en el área de desarrollo de sistemas automatizados para la identificación de personas, los cuales obtienen datos biométricos específicos de cada sujeto. Un dato biológico es cualquier valor obtenido de las propiedades biológicas y fisiológicas y cualidades particulares de una persona; concluyendo que un dato biométrico es único y singular porque pertenece a una sola persona. Por tanto, es por esta razón que es muy común el uso de algún dispositivo de tipo biométrico en diferentes entidades para la identificación de personas. Dentro de los sistemas biométricos que son los convencionales están la detección por la huella, iris ocular, voz, facial, contornos de las manos, etc. Sin embargo, hay casos en los que algunas de estas medidas biométricas han sido clonadas para usos ilegales, donde las más propensas a la clonación son la huella dactilar y la voz. Así, un informe de la BBC de 2014 menciona que en un discurso presentado por Jan Krissler en la convención del Chaos Computer Club, que es una comunidad europea de piratas informáticos más grandes de Europa, afirmó haber clonado la huella dactilar del ministro de defensa alemán, simplemente utilizando fotografías tomadas durante una charla pública. Una solución para combatir los desafíos de la clonación de huellas dactilares es el uso de hardware que emplea la tecnología LFD - Live Finger Detection, que se desarrolló para evitar el acceso a los sistemas de seguridad cuando se intenta ingresar utilizando huellas dactilares falsas hechas de látex, plástico, caucho, etc. Sin embargo, esta solución no es económicamente viable para las pequeñas empresas debido a su alto costo. Los sistemas de identificación biométrica mencionados anteriormente han sido evaluados, de acuerdo con sus principales características, como se muestra en la revista tecnológica Agarwal y Verma en la que se afirma de su estudio que la detección de huellas de palmas sería la mejor manera de lograr la identificación y protección de una persona.

.

El presente tema de investigación parte de la identificación de un problema, en donde las empresas dedicadas a la construcción y venta de departamentos en Lima. En la actualidad los trabajadores de estas empresas presentan deficiencias en el marcado de asistencia laboral, el cual se realiza mediante un equipo biométrico basado en la huella dactilar. Debido a la labor que realizan los obreros sus huellas dactilares son borrosas, provocando múltiples marcaciones negativas y la demora del ingreso del personal a la empresa, trayendo como consecuencia la insatisfacción y retraso en sus labores cotidianas asignadas. Tomando en cuenta esta realidad y buscando una posible solución, se ha creído conveniente la investigación de un nuevo método de identificación biométrico que mejorarán de manera significativa esta situación. Dado que aparte de tener un lugar más seguro y confiable para la empresa, el personal utilizará tecnología de vanguardia y así cumplirá con sus metas marcadas. Es imprescindible resaltar que los dispositivos integrados están ganando mayor atención en biometría debido a la confiabilidad y los sistemas rentables. (Guarav & Amit, 2018, págs. 53-78)

En el contenido de la investigación se han tomado los temas de seguridad empleados en la actualidad, la descripción y evaluación de los sistemas biométricos más usados a nivel mundial. Considerando que Palmprint se define como el patrón de línea que se encuentra dentro del área de la palma, demostrando que Palmprint se distingue de otras características debido a una serie de atributos, estos atributos incluyen color, claridad, posición, continuidad, longitud y variación de espesor y teniendo en cuenta el adecuado balance de costos y beneficios, es necesario el estudio de nuevos métodos biométricos basados en las huellas palmares con la finalidad de dar una posible solución, que en la actualidad existen; no obstante al tener en cuenta la problemática donde los trabajadores tienen las huellas dactilares borrosas por la labor ruda que realizan a diario, éstas se van deteriorando, generando así marcaciones negativas. Es por ello que implementando el sistema de edificación mediante palmprint, se lograría controlar los horarios de los empleados para obtener un mejor desempeño y a su vez mejorar la

productividad de la labor que desempeñan en las empresas constructoras. (Bertino & Orozco, 2018, págs. 18-32)

Se tomarán como grupo objetivo las huellas palmarias de los trabajadores de una empresa constructora, que cumplan los criterios de inclusión para el estudio, resaltando las características palmarias de dichos trabajadores; debido a la actividad que realizan en la empresa constructora son lesionadas al transcurrir los días; por ende, es importante implementar un equipo biométrico que analice las huellas palmarias lesionadas, con la finalidad de lograr la identificación de cada personal. El nuevo método de identificación basado en las impresiones palmarias mejorará el control biométrico actual del personal de la constructora. (Bertino & Orozco, 2018, págs. 18-32)

El presente trabajo de investigación presenta una iniciativa para desarrollar un método alternativo de identificación eficiente para identificar a cada individuo, mediante un adecuado pre procesamiento digital de imágenes en tiempo real de impresión palmaria, utilizando un escáner en diversos horarios del día. Se pretende conseguir como resultado un método confiable. (Blanco, 2018, págs. 5-28)

Figura 1. Palma de la mano Izquierda borrosa y con afectaciones de obrero.

Fuente: Elaboración Propia.

1.2. Antecedentes de estudio

Gopal, Srivasta, Bhardwaj, & Bhargava, (2016), realizaron la investigación, *Fusion of impression of palmeral phalanges with palmar impression and dorsal vein of the hand*, en el Instituto de Tecnología Netaji Subhas, Delhi, india. Para aumentar el rendimiento de un sistema biométrico, en el presente estudio se combinaron varios sistemas unimodales para evitar sus limitaciones utilizando un sistema biométrico multimodal. Aquí se presentó un sistema de autenticación personal multimodal que utiliza la huella de la palma, el patrón de la vena dorsal de la mano y una novedosa modalidad biométrica "impresión de las falanges de la palma". El estudio presenta una nueva base de datos de manos anteriores de 50 personas con 500 imágenes en el instituto denominado NSIT Palmprint Database utilizando el dispositivo de impresión palmar NSIT. Luego a partir de estas imágenes de la mano anterior, se implementaron bases de datos para la impresión de la palma y las falanges de la palma. El estudio determina que, haciendo uso de este sistema biométrico, las personas no tienen incomodidad para usar dos sensores diferentes. Debido a que las características de impresión de la huella de la palma y los dedos de las manos se pueden registrar a partir de la misma imagen, empleando el dispositivo de impresión de la palma NSIT, en paralelo. También para la vena dorsal de la mano, Bosphorus Hand Vein Database fue utilizada debido a la estabilidad y singularidad de los patrones de las venas de la mano. El estudio propone la fusión de tres modalidades biométricas diferentes que incluyen la impresión de la palma (PP), la impresión de las falanges palmas (PPP) y la vena dorsal de la mano (DHV) y realizaron una fusión a nivel de puntuación de PP-PPP, PP-DHV, PPP-DHV y PP-PPP como estrategias de DHV. El estudio prueba que la fusión multimodal presenta ventaja sobre la fusión unimodal. Por último, se concluye en el estudio el valor K más cercano, la máquina de vectores de soporte y el bosque aleatorio para validar la etapa de coincidencia.

Shashi & Nidhi, (2016), realizaron la investigación. *Comparative analysis of the palm print recognition system with the repeated line tracking method.* en el National Institute of Technology, Kurukshetra, India. La técnica Palmprint presenta variadas características con una serie de atributos como el color, continuidad, claridad, longitud, posición, y variación de espesor. Sin embargo, presenta inconvenientes de líneas rotas o faltantes cuando se realiza el proceso de lectura y registro de la huella de la palma, generándose complicaciones en el proceso de emparejamiento. Para descartar este inconveniente en las imágenes binarias, el estudio propone una técnica eficaz para disminuir la cantidad de líneas repetidas y faltantes. El estudio menciona que la técnica de mejora para el reconocimiento de la huella palmar, proporciona una puntuación y precisión de coincidencia eficiente. Así también el estudio propone que las líneas son patrones altamente efectivos para la representación de formas. Finalmente, las líneas se muestran de una manera muy eficiente e inclusive ocupando un menor espacio para su almacenamiento, con una consistencia en la detección y eficiencia en la coincidencia de formas que conforma una gran base de datos. Asimismo, para lograr reducir el tiempo de respuesta y obtener una puntuación de 100% es conveniente aplicar la técnica de seguimiento de líneas repetidas.

Aashni, Archanasri, Nivedhitha, Shristi, & Jyothi, (2017), realizaron la investigación, Hand Gesture Recognition for Human Computer Interaction, en el departamento ingeniería, BMS colegio de ingeniería, Bangalore, India. Durante la implementación del sistema de reconocimiento eliminado el fondo, los autores encontraron varias dificultades de precisión, debido a que la eliminación de fondo no favorece al realizar modificaciones de iluminación radicales y súbitos que generan muchas inconsistencias. Además, el método prescinde de una selección exhaustiva de parámetros. Debido a estas complicaciones enfrentadas el estudio propone utilizar contornos, defectos de convexidad y cascada de Haar para detectar el objeto. El estudio brinda un sistema de reconocimiento de gestos robusto sin el uso de marcadores, haciéndolo fácil de utilizar a un menor costo. En este sistema de

reconocimiento de gestos, se ha intentado proporcionar gestos, que cubren casi todos los aspectos de HCI (Human Computer Interaction), como las funcionalidades del sistema, el desarrollo de aplicaciones y el uso de algunos sitios web populares. En el estudio los métodos que se utilizaron para capturar la información del usuario son las bases de datos y las cámaras. Como también para la lectura de los movimientos se emplearon bluetooth, giroscopio y medidores de aceleración. Para los registros de colores y profundidad utilizaron los autores cámaras senz3D y estéreo Bumbleblee2. Modelos rentables como ICCE-TW, INDICON y VSL han implementado sus sistemas utilizando cámaras web simples. Con el soporte de cámaras RGB de profundidad de Kinect, los autores aplicaron el método ROBIO logrando capturar la corriente de color proporcionándoles información de profundidad detallada para cada pixel. En el primer conjunto de evaluación, el estudio propone entornos que contenían diferentes tipos de fondos planos sin ninguna inconsistencia. En una segunda prueba se emplearon fondos con diferentes inconsistencias. Definiéndose como la precisión del resultado al promedio de la cantidad de veces que se logró un reconocimiento correcto de un gesto. Se notó la funcionalidad del reconocimiento de gestos toda vez que se cuente con un fondo plano, sin inconsistencias y de color liso.

Lo que demuestra que para producir los mejores resultados posibles y una gran precisión es conveniente usar para el sistema reconocimiento de gestos un fondo plano. Por otro lado, el estudio focaliza maximizar escenarios de dominio con procesos de seguimiento en hardware en equipos digitales para todo tipo de usuarios tales como personas con discapacidades.

Maeda, et al, (2018), realizaron la investigación, *Convolutional neural networks for detection and classification of plant diseases based on digital images*, en la Universidad Autónoma de Zacatecas, México. Las enfermedades de plantas representan factores adversos que causan una grave reducción en la calidad y cantidad en cultivos agrícolas, es por ello que los autores utilizaron técnicas de inteligencia artificial y visión computacional para desarrollar un autómata capaz de clasificar las enfermedades de las hojas. Los autores evaluaron diversas arquitecturas para clasificar las

enfermedades y obtuvieron variaciones entre 98.90% y 96.22% de exactitud, donde la arquitectura de AlexNet, entrenada con un conjunto de datos PlantVillage-Dataset, obtuvo el porcentaje más alto. A partir de ello, encontraron que el Deep Learning permite extender las investigaciones, pero dependiendo del objetivo el uso de las arquitecturas puede variar; a pesar de ello se halló que la arquitectura de AlexNet era la arquitectura que obtenía mejores resultados en comparación con otras arquitecturas tales como SqueezeNet y GoogleNet. Los autores concluyeron que es necesario utilizar las CNN (Redes Neuronales Convolucionales) con la finalidad de crear una herramienta para que los investigadores busquen diseñar algún autómata para clasificar las enfermedades de hojas de plantas, ya que puede brindar información acerca de la arquitectura que puede ser más conveniente.

Dalia, Aboul, & Hassan, (2020), realizaron la investigación, *A Deep Learning Architecture Optimized for COVID-19 Disease Diagnosis Based on Gravitational Search Optimization,* en la Universidad de El Cairo, Egipto. Con imágenes de rayos X de tórax que permite obtener un nivel alto de precisión para los diagnósticos de COVID-19, al cual le denominaron GSA-DenseNet121-COVID-19, por medio de un optimizado algoritmo para el desarrollo de una red neuronal híbrida (CNN). Para conseguir valores óptimos para los hiperparámetros de la arquitectura DenseNet121, el estudio determinó un algoritmo de búsqueda gravitacional (GSA). Dando como resultados al 98.38% del total de pruebas. El estudio también compara el GSA versus el SSD-DenseNet121 siendo este último dependiente del DenseNet121 y del algoritmo denominado controlador de esqui social (SSD). Los resultados de la comparación demostraron la eficacia de la propuesta GSA-DenseNet121, el segundo pudo diagnosticar solo el 94% del conjunto de prueba. Los resultados de las comparaciones demostraron según el estudio que GSA-DenseNet121-COVID-19 logró superar y por tanto es altamente competitivo.

Gaurav, Amit, & Ravinder, (2018), realizaron la investigación, Multifunctional *Fusion for Unrestricted Palm Print Authentication, Biomedical Instrumentation and Signal Processing Laboratory*, en el Instituto Nacional de Tecnología, en la India. Este artículo muestra la unión de características de matriz de ocurrencia de código de textura, transformación e invariantes de escala, logrando un reconocimiento efectivo. El estudio enfatizó en la definición de la imagen de la zona de interés extraída, seguido por la aplicación de la máscara diferencial fraccionaria que mejora el detalle de la textura. Un algoritmo de transformación permitió seleccionar los caracteres de la palma más discriminada basado en el aprendizaje sub espacial, consiguiendo la disminución del tiempo de lecturas y funciones y por consiguiente un buen registro. El estudio menciona que, mediante un análisis experimental comparativo descrito en este documento, los resultados fueron superiores a los métodos competidores, consecuentemente validaron la eficacia del enfoque propuesto.

Hafiz, et al, (2019), realizaron la investigación, *Automated identification of fish species based on visual characteristics using, deep convolutional neural networks*, en la Universidad de Gujrat en Departamento de Computación, Pkistán. En cuanto a la identificación de especies de peces basada en la morfología, es debido al extenso tiempo que se requiere para su proceso, ya que existen numerosas especies de peces y debido a su gran parecido entre sí, es difícil clasificarlos por caracteres externos. En el estudio mencionaron que los investigadores están utilizando ampliamente la visión por computadora y la identificación basada en el aprendizaje profundo de diferentes especies animales. El estudio propone la red neuronal convolucional (CNN) como una de las herramientas analíticamente más poderosas en la arquitectura de aprendizaje profundo para la clasificación de imágenes basada en características visuales, en un marco de aprendizaje profundo basado en el método CNN para la identificación de especies de peces. El método CNN presenta 32 capas profundas que permiten en la imagen determinar características particulares y discriminatorias. La VGGNet para aumentar el rendimiento de clasificación utilizó una profunda

supervisión, adicionando durante el entrenamiento cuatro capas convolucionales a cada nivel en la red. Como prueba de eficacia del CNN de 32 capas propuesta, desarrollaron un conjunto de datos denominado Fish-Pak que contiene 915 imágenes con seis clases distintas; carpa herbívora, *Cyprinus carpio*, *Labeo rohita* (Rohu), carpa plateada y Thala y tres vistas de la forma del cuerpo, la escala y de la región de la cabeza. Para garantizar el rendimiento superior de la arquitectura CNN propuesta, el estudio llevó a cabo la contrastación experimental con otros marcos de aprendizaje profundo que involucran VGG-16 para el aprendizaje de transferencia, un bloque VGG, dos bloques VGG, tres bloques VGG, LeNet5, AlexNet, GoogleNet y ResNet50 en el conjunto de datos Fish-Pak. Finalmente, los análisis empíricos completos revelaron que el método propuesto logró un rendimiento de vanguardia y supera a los métodos existentes.

Poonam, Pawan, & Vijayendra, (2020), realizaron la investigación, *Palmprint Recognition using Robust Template Matching*, Birla instituto tecnológico de Pilani, India. La información de las minucias y su ubicación no están disponibles en la planilla, lo que hace imposible estimar la orientación de la huella de la mano a partir de la plantilla. La presente investigación propone una plantilla de impresión palmar no invertible que almacena la información sobre la orientación geométrica relativa de los puntos minuciosos. El estudio menciona que para la coincidencia de plantillas utilizaron en el estudio la coincidencia de ángulos internos basada en la triangulación de Delaunay, que es computacionalmente eficiente. El método propuesto se prueba en bases de datos estándar PolyU, IIT-Delhi y Multi-espectral palm-print y según el estudio produjeron mejores resultados en términos de tasa de reconocimiento correcto (CRR) y tasa de error igual (EER) de 95,4% y 0.37% respectivamente.

Mejía, Antón, Flores, Tuesta, & Forero, (2020), realizaron la investigación, *New method identification based on palmprint*, en el Laboratorio de Sistemas inteligentes y seguridad informática, Facultad de Ingeniería de Sistemas, Universidad Señor de Sipán, ciudad Chiclayo, en el país de Perú. Puesto

que los fraudes de personas inescrupulosas han logrado vulnerar los sistemas conocidos de detección de caracteres de huellas, rostros, tonos de voz y retinas oculares. Por tanto, el estudio enfatiza que los usos de datos están desarrollando nuevos sistemas más eficaces, infalibles y rápidos para identificar a cada persona, haciendo imposible para hacerse pasar por ellos, como ha sucedido con otros métodos. Incorporando la detección de la geometría de la mano, impresión de la palma con registro de características particular la técnica de contacto para la detección de palmaria con registros de imágenes a través de un escáner. Delimitando y acortando la zona de detección, recortando la imagen mostrando solo la silueta de la mano para su detección, mejorando los tiempos de respuestas para la identificación. Estas imágenes son ingresadas a una red neuronal convolucional VGG 16 para el aprendizaje e identificación de sujetos, con una precisión del 100%.

Tuesta, Alcarazo, Mejía, & Forero, (2020), realizaron la investigación, *Clasificación automática de Limón en base del procesamiento de imágenes digitales*, en el Laboratorio de Sistemas Inteligentes y Seguridad Informática, Facultad de Universidad Señor de Sipán, en la ciudad de Chiclayo en el país de Perú. La falta de tiempo para la selección y clasificación de los limones, Citrus *Aurantifolia* swingle, el cual es cultivado en la costa norte de Perú para consumo interno y exportación, provocando que los limones se clasifiquen erróneamente con frecuencia debido a la falta de concentración, agotamiento y experiencia del trabajador, afectando la calidad del producto vendido en el interior y el mercado exterior. Esta selección se hace de forma manual. El artículo describe un nuevo método para la clasificación automática de *Citrus Aurantifolia*, el cual considera la adquisición, extracción de características y la clasificación; un patrón mecánico para la adquisición de imágenes y un software para la clasificación de limones. La técnica considera la implementación del método de segmentar, con información del canal azul, dando como resultado la obtención de las características de color, en los espacios RGB y CIELAB, dando como resultado que el canal rojo permite la mejor precisión frente a los modelos de clasificación, SVM y KNN, obteniendo el K-NN una mejor precisión del 99,04%

Jingle, et al, (2020), realizaron la investigación*, Analyze the impact of soft errors on VGG networks implemented on GPUs*, Research Institute, China. Este articulo describe la efectividad del modelo VGG, puesto que uno de los inconvenientes es justamente las fallas que puede tener un software. El modelo contiene una arquitectura CNN, a través las métricas del Factor de Vulnerabilidad de Kernels (KVF) permitiendo identificar kernels vulnerables y el Factor de Vulnerabilidad de Capa (LVF) para determinar la propagación de fallas de pista a través de capas, ejecutado en una GPU de NVIDIA, utilizando el inyector de fallas llamado SASSIFI como principal herramienta de evaluación. Este análisis muestra que Im2col presenta la mayor vulnerabilidad entre todos los kernels y el fortalecimiento de este kernel disminuye la tasa de corrupción de datos silenciosos en un 85,67% con una penalización de tiempo del 35,63%. El análisis menciona que el promedio de una clasificación errónea llega hasta el 19,7% en algunos modos de inyección, por tanto, inaceptable. Se propone que el modo de inyección de fallas afecta a KVF y a LVF porque el modo de RF se realiza a nivel de arquitectura, mientras que los modos IOV e IOA están en un nivel de programa. Se efectúa una comparación entre VGG, AlexNet y ResNet, en donde LVF y KVF cuentan con la arquitectura de red de CNN, puesto que las capas de estas redes, específicamente las capas convolucionales, fueron implementadas con el más vulnerable, el kernel *Im2col*.

1.3. Teorías relacionadas al tema.

1.3.1 Visión por computador

La visión por computadora está definida por la ciencia interdisciplinaria encargada de entender cómo las computadoras obtienen un alto nivel de compresión a partir de videos digitales o imágenes digitales. Desde la mirada de la ingeniería se desea comprender y automatizar el sistema visual humano.

Son tareas visuales por computadora las que abarcan técnicas para conseguir, analizar, comprender imágenes, procesar y adquirir datos con una dimensión alta sobre mundo real y proporcionar información numérica La comprensión en esto significa la modificación de las imágenes visuales o de la entrada de la retina que describe el mundo que tienen sentido para los procesos de forma de pensamiento y pueden provocar la acción acertada. Este entendimiento de imagen se puede ver como la segmentación de la información con símbolos de los atributos de la imagen, empleando los modelos, son construidos con ayuda de la física, la estadística, la geometría, y la definición del aprendizaje.

1.3.2 PalmPrint

Una impresión de palma o palmprint es una imagen digital adquirida de la región de la palma de la mano. Podría ser una imagen digital tomada por medio de un escáner, como también con una imagen fuera de línea es decir tomada con tinta y papel. La palma consta de crestas epidérmicas, líneas principales y líneas secundarias a las cuales denominamos arrugas. Difieren de las huellas dactilares puesto que contienen más información como marcas, texturas, sangrías que permiten una comparación de palmas. (Rodes, 2018, págs. 17-30). La captura de la palma de la mano puede ser aplicada para actividades de registros forenses, puesto que frecuentemente los malhechores dejan huellas de las palmas durante sus hechos delictivos a pesar de usar guantes, puesto que no llegan a cubrir la totalidad de la mano o se sale la mano de parte del guante dejando expuesta parte de las palmas y por tanto dejando sus rastros. (Rodes, 2018, págs. 17-30)

Figura 2. Imagen de la impresión palmaria de la mano derecha

Elaboración Propia.

1.3.3. Biometría

La biometría agrupa las mediciones y cálculos corporales relacionados con los atributos del individuo. Es necesario tener en cuenta que la autenticación biométrica es aplicable en informática para el control de acceso y como un tipo de identificación, además para la identificación de personas agrupadas que se encuentran bajo supervisión. La biometría caracteriza distintivamente de forma medible para tipificar y reconocer a las personas. Los equipos biométricos por lo general se catalogan teniendo en cuenta las características físicas o de comportamiento, las físicas están relacionadas con la geometría del cuerpo, como también venas de la palma, reconocimiento facial, ADN, huellas, identificación del iris y/o retina, aroma y geometría de la mano, mientras que las de comportamiento van relacionadas con patrones de comportamiento de la persona, como la marcha, el ritmo de mecanografía y la voz. Varios profesionales especialistas describen a la conductimétrica cuando explican de la biometría. Las técnicas más típicas para el control de acceso consideran un sistema de identificación con registros personales, como un pasaporte o brevete, y sistemas de identificación apoyado en el conocimiento, como el número de documento nacional de identificación o una clave. En comparación con los métodos que utilizan token o conocimiento, los equipos de biometría aplican para

individuos u objetos de forma más segura; no obstante, aún presenta cuestionamientos sobre su uso adecuado puesto que contiene información personal privada de cada individuo. (Sancho, 2018, págs. 19-33)

1.3.4. Autenticación Biométrica

Es la comprobación de la identidad de individuos por medio de un dispositivo y software. Lo cual lo distingue del hecho de indicar la identidad de las personas u objetos, el proceso de autenticación confirma la identidad. Permitiendo verificar la autenticidad de los documentos personales de identificación, perfectamente utilizado para descartar los documentos fraudulentos de identificación. (Sancho, 2018, págs. 19-33)

1.3.5. Aprendizaje supervisado

Deduce una función con datos de entrenamiento, su meta es aprender fórmulas generales que mapean entradas y salidas. (Sancho, 2018, págs. 19-33)

1.3.6. Redes neuronales

El gran desarrollo que está viviendo la tecnología asociada con la inteligencia artificial (IA) en los últimos tiempos se está dando nuevas y novedosas aplicaciones.

Una de las áreas donde existe avance es en el reconocimiento de imágenes, Deep Learning o aprendizaje profundo.

Una red neuronal procede del pensamiento de emular la actividad de las redes neuronales de todo ser, mediante una reunión de neuronas enlazadas entre sí trabajando en equipo, sin existir una actividad determinada para cada una.(Gampala, Gurugubelli, & Hari, 2018, págs. 91-100)

1.3.6.1. Ventajas de las redes neuronales artificiales

Las redes neuronales del tipo artificiales contienen características muy parecidas a las del cerebro. (Rubio, Hernández, Ávila, Stein, & Meléndez, 2016, págs. 211-222) Las ventajas son:

- Tolerancia a fallos.
- Operaciones en tiempo real.
- Aprendizaje adaptativo.
- Autoorganización.
- Fácil inserción dentro de la tecnología existente.

1.3.7. Red Neuronal Artificial

Las redes neuronales artificiales (ANNs), llamadas en general como redes neuronales (NNs), definido por sistemas informáticos acorde con la funcionalidad que las redes neuronales biológicas de los cerebros. Una red neuronal artificial se fundamenta en una agrupación de unidades enlazadas denominadas neuronas artificiales las que modelan autónomamente las neuronas en un cerebro. Similar a las sinapsis en un cerebro biológico, una conexión de neuronas puede transferir una señal a otras neuronas, de tal forma que la neurona que recepciona, la procesa y enseña a las conectadas entre sí. Con un valor numérico se define a una señal y la información de salida ingresa a un cálculo con la sumatoria de las entradas en una función no lineal. A las conexiones se les denominan aristas, los bordes y las neuronas usualmente llevan un peso acorde conforme avanza el aprendizaje. El peso puede disminuir o aumentar la fuerza de la señal en una conexión, pueden alcanzar un umbral de tal forma que una señal se direccione sola, si la señal agregada cruza ese umbral. Las neuronas se agrupan en capas, que pueden realizar diversas variaciones en sus entradas, logrando transferir las señales desde la capa de entrada y la capa final. La arquitectura ANN utilizada para el modelado se muestra en la Fig.2 con cuatro variables de entrada, tres capas ocultas respectivamente y el desgaste volumétrico como variable de salida (Gampala, Gurugubelli, & Hari, 2018, págs. 91-100)

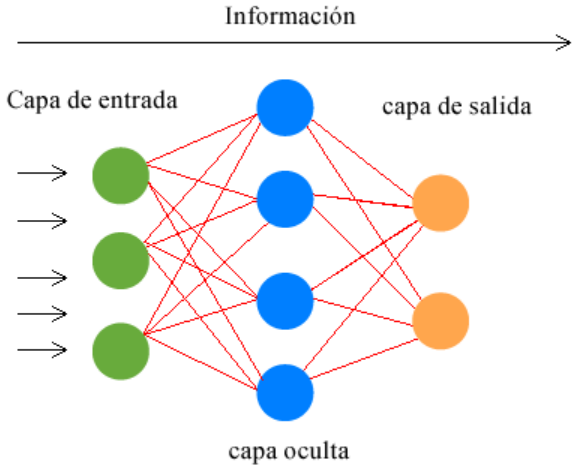

Figura 3. Forma de trabajo de la arquitectura de una Red Neuronal

Fuente: Elaboración Propia.

1.3.7.1. Red Neuronal Convolucional

Una red neuronal convolucional nació con la necesidad de poder procesar imágenes. Catalogado como una muestra de red neuronal artificial en el cual las neuronas forman parte de campos receptivos, semejantes a las neuronas un cerebro biológico. Una red neuronal convolucional es el trabajo de un perceptrón de multicapa, donde debido a los usos que se les da, son realizadas en matrices binarias y bidimensionales, en las cuales se evidencia lo efectivas que son éstas para propósitos de la visión artificial en la segmentación y clasificación de imágenes, como entre otras muchas aplicaciones.

La red neuronal convolucional tiene una capa de entrada y también cuenta con una de salida, con diversidad de capas ocultas, las cuales suelen consistir en una serie de capas convolucionales que se convolucionan con una multiplicación u otro producto escalar. Una capa RELU es la función de activación, y luego se continúan las convoluciones adicionales como lo son las capas de agrupación, capas conectadas en su totalidad y capas ocultas o también llamadas normalización por sus entradas y salidas tienen máscaras por funciones de activación y la capa de convolución final, donde las capas son llamadas comúnmente como las convoluciones, y se dice esto por acuerdo. Es un producto matemáticamente hablando, donde el producto

de puntos es de relación cruzada y esto nos interesa para los índices matriz, esto afecta el modo en que se define los pesos en un sitio indicado del índice.

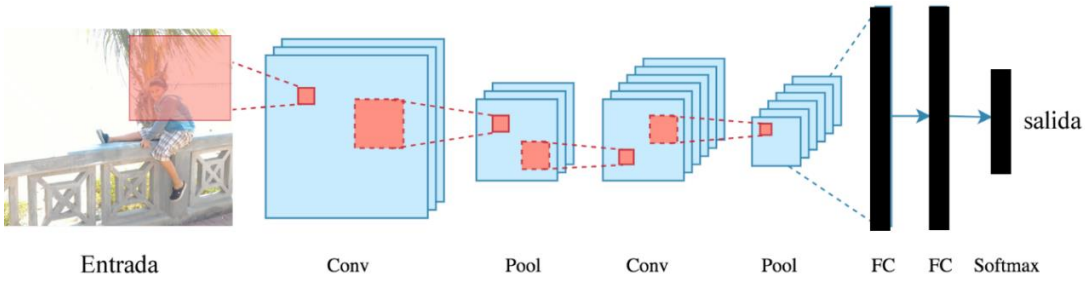

Figura 4. Procesos de una red Neuronal Convolucional de machine learnig

Fuente: Elaboracion propia

1.3.7.2. Neuronal Convolucional

En la etapa de adquisición de características cada neurona convolucional más común es sustituida por los procesadores donde la matriz que corre acción sobre la información de las imágenes en matrices 2D, en vez de un valor de número donde la fórmula de cada neurona convolucional en la capa de salida es el siguiente:

$$Y_j = g(b_j + \sum_i Kij\, Y_i)$$

Donde una neurona en salida es la Y_j y j calcula por la combinación lineal por ser una matriz, Y_i cada una de las neuronas operadas lo son con el núcleo de la red neuronal convolucional K_{ij} y ésta pertenece a esa conexión. El número o cantidad se sumará a la influencia b_j la cual será pasada por la función de activación $g(.)$ que es de tipo no-lineal.

El programador podría buscar la imagen que usara en la entrada con un modelo que fue anteriormente entrenado. Ello genera un cambio en los datos dado que algunas características, basadas en el modo del núcleo y éstos transforman en dominantes de la imagen de la salida al presentar un alto valor que previamente se asignó a los píxeles. Los núcleos cuentan con la

destreza en el procesamiento de imágenes concretos, como la identificación de bordes realizados con núcleos que enfatizan la gradiente a determinadas direcciones. Los núcleos son adiestrados generalmente por una red neuronal convolucional cuentan con mayor complicación para poder extraer otros atributos abstractos y no triviales. (Gampala, Gurugubelli, & Hari, 2018, págs. 91-100)

1.3.7.3. Neurona de Reducción de muestreo

Las redes neuronales toleran escasas alteraciones en la data de entrada; donde dos imágenes parcialmente semejantes (se diferencia por el traslado de ciertos píxeles lateralmente) son analizados utilizando una red neuronal, cuyo resultado debe ser el mismo. Se alcanza por la disminución de muestreo, esto sucede por dentro de una red neuronal. Al momento de reducir la resolución, los mismos atributos corresponden a una mayor imagen de entrada.

En sus inicios las redes neuronales convolucionales usaban una serie de pasos de subsampling que permiten realizar esta operación. No obstante, nuevos estudios demostraron que otras sistematizaciones, como el max-pooling, tienen mayor eficacia en sintetizar atributos en un área. Asimismo, hay evidencia que este tipo de operación es análoga al proceso de síntesis de información de la corteza visual.

Max-pooling busca un valor máximo entre una ventana de muestra y entrega valor en forma de características en la determinada área, trayendo como resultado la disminución del tamaño de datos siendo igual el tamaño de la ventana de muestra en la cual trabaja. (Gampala, Gurugubelli, & Hari, 2018, págs. 91-100)

1.3.7.4. Neurona de clasificación

Posterior a la selección de atributos, los datos convergen a la etapa de clasificación. Por ello, fueron escogidos los datos de hasta una sucesión de

atributos únicos para la imagen de entrada, el objetivo de esta última etapa es el clasificar estos atributos hacia una u otra etiqueta, teniendo en cuenta los fines de entrenamiento. En esta etapa las neuronas de clasificación funcionan de una manera semejante a las de un perceptrón de multicapas, donde cada salida es calculada mediante la siguiente ecuación:

$$Y_j = g(b_j + \sum_i w_{ij} Y_i)$$

Donde y_i es la salida de la neurona j es un valor calculado mediante la mescla lineal de output y_i la neurona para la anterior capa se multiplica por un peso w_{ij} y corresponde a la conexión. Esto es sumado b_j y en seguida se pasará por activación $g(.)$ no-lineal.

1.3.8. Imagen.

Dubuisson, citado por (Forero, 2002) manifiesta que se puede definir una imagen bidimensional estática, como una representación en un plano de una escena o de un objeto situado, en general, dentro de un espacio tridimensional. Se obtiene cada vez que un sensor es excitado por los rayos que han interactuado con los objetos. El modelo del proceso puede ser visto desde diferentes apreciaciones (Forero, 2002, pág. 11):

a. Modelo abstracto.
b. Modelo geométrico de transformación de escena 3D sobre un plano 2D.
c. Modelo óptico.
d. Modelo espacial.
e. Modelo del comportamiento espectral.
f. Modelo digital.

Este último aparece con la necesidad de utilizar herramientas computacionales en el procesamiento de imágenes. En este tema, una imagen continua o analógica, que representa un objeto o una escena dada, al ser modelizada mediante una digitalización, se traduce en una serie de muestras

igualmente espaciadas formando un arreglo bidimensional, donde a cada elemento corresponde una amplitud asociada a un color, intensidad luminosa o nivel gris
(Forero, 2002, pág. 11).

1.3.9. Imagen digital

La imagen digital es f (x, y) la cual se le ha separado tanto en el brillo como en coordenadas espaciales. Es la representación para cada banda de color por una serie de matrices en 2D. El valor de luminosidad digitalizada se llama nivel de gris, los elementos de la matriz son denominados como pixel, deriva de la expresión “elemento de imagen”. Por lo general, el tamaño de una matriz de este tipo es unos pocos cientos de píxeles por unos pocos cientos de píxeles y hay varias docenas de posibles diferentes niveles de gris (Petrou & Petrou, 2010, p. 1).

Por lo tanto, una imagen digital se representa de la siguiente forma:

$$f(x,y) = [f(1,1)\ f(2,2) \quad \cdots \ f(1,N)\ f(2,1)\ f(2,2) \quad \ldots \ f(2,N) \ \vdots\vdots \\ \vdots \ f(N,1)\ f(N,2) \quad \ldots \ f(N,N)\ ]$$

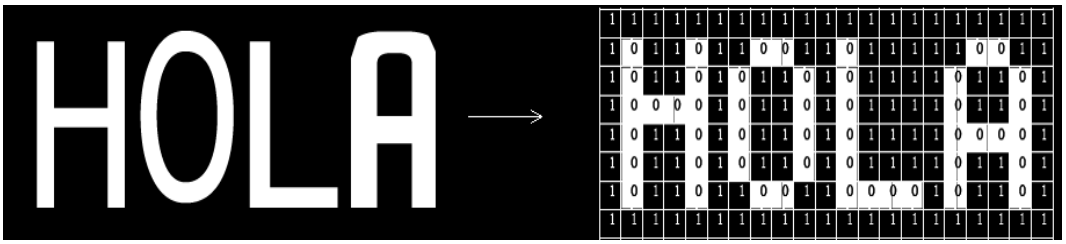

Figura 5. Comparación de una Imagen digital e imagen digital en Matriz binaria
Fuente: Elaboración Propia

1.3.9.1. Calidad de la imagen

Es un concepto complicado, en gran medida subjetiva y muy dependiente de la aplicación” (Petrou & Petrou, 2010, p. 7). Tener una imagen con buena calidad

implica que la imagen no tenga ruido, que sea clara, que tenga alta resolución y un buen contraste.

1.3.9.2. Pixel

Está definido como el elemento direccional diminuto de una imagen de los dispositivos de visualización; controlable y representada en la pantalla. Cada píxel es la muestra de una imagen digital original que por lo general son ceros y unos; existe una relación proporcional entre el mayor número de muestras y la precisión de las representaciones del original

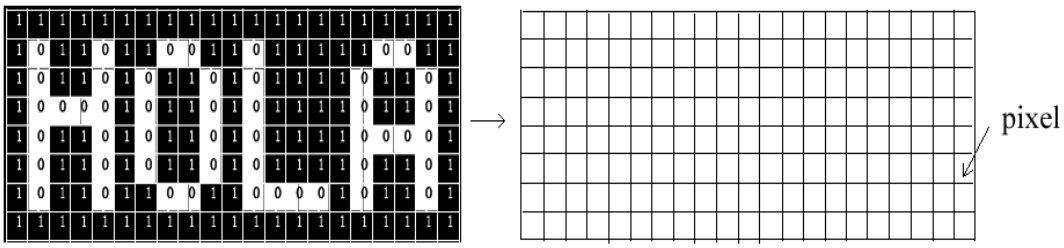

Figura 6 Ejemplo de representación de un píxel en el cartel hola.

Fuente: Elaboración Propia.

1.3.9.3. Imagen borrosa

La imagen borrosa es causada por las condiciones de captura de imágenes de manera incorrecta; es decir debido a cámara desenfocada o el movimiento rotativo de la cámara y el objeto fotográfico. La cantidad de desenfoque de la imagen es expresada por la llamada función de dispersión del sistema de la imagen (Petrou & Petrou, 2010, p. 7).

1.3.10. Dataset

Es la recolección de información o datos como información tabular, un conjunto de datos corresponde a tablas de base de datos, donde cada tabla y su columna representan una variable diferente, y cada una de las filas es de un registro dado del conjunto de datos. Esta cantidad de datos enumera las

equivalencias para cada una de las variables, como el peso de un objeto y su altura, para todas las partes del conjunto de datos. A toda equivalencia se le dice que es un dato. Asimismo, hasta los conjuntos de los datos serían una colección de archivos o documentos. En el método los datos abiertos forman parte del conjunto de datos y son el mecanismo para calcular los resultados publicados en un repositorio de datos libres. Algunas otras definiciones para dataset podrían ser fuentes de datos en tiempo real o un conjunto de datos no relacionales que aumentan la dificultad para llegar a un consenso al respecto en las aplicaciones.

1.3.10.1. Digitalización de imágenes

Digitalizar las imágenes abarca una serie de pasos que va desde transformar fotografías a un formato digitalizado, en donde la información es ordenada en bits.

El resultado es denominado representación digital. Actualmente los datos digitalizados se connotan en números binarios, facilitando de esta manera el procesamiento por computadora, cabe resaltar que digitalizar es la transformación de una fuente analógica hasta llegar a una numérica u otro sistema numérico que se usará en su defecto.

1.3.11. Procesamiento de Imagen.

Es la asociación de prácticas que permiten transformar las imágenes para un acrecentamiento de perceptibilidad de sus atributos, para después verificar con un test, o un crecimiento de visualización de la imagen digital de modo que no incrementa la data, la cual permite sacar de las imágenes sus características, en otras palabras, es el realce de atributos de las imágenes digitales para su extracción. (Viera, 2018)

El procesamiento de imágenes digitales puede realizarse a través de métodos visuales o digitales en un computador (Vacacela, Fiama, & Mantilla, 2015)

1.3.11.1. Pre-Procesamiento de Imágenes

El preprocesamiento abarca una serie de pasos aplicados a las imágenes de intensidad de entrada y salida. El preprocesamiento de imágenes es un método para transformar los datos de imágenes sin procesar en datos de imágenes limpias, ya que la mayoría de los datos de imágenes sin procesar contienen ruido y algunos valores faltantes o incompletos, valores inconsistentes y valores falsos. El objetivo del preprocesamiento es mejorar los datos de la imagen para disminuir las falsificaciones reacias o para perfeccionar algunos atributos de la imagen prioritarios para un procesamiento adicional. No obstante, la aplicación inteligente del preprocesamiento de imágenes puede ofrecer beneficios y solucionar problemas que finalmente producen una mejor detección de las funciones globales y locales.

El preprocesamiento de imágenes puede tener efectos beneficiosos sobre la excelencia de la extracción de características y los resultados del análisis de imágenes. (Vacacela, Fiama, & Mantilla, 2015)

1.3.11.2. Filtro de una imagen.

El principal objetivo del procesamiento es mejoramiento de las imágenes, abarca una agrupación de técnicas tendientes a optimizar la calidad visual de una imagen. Estas operaciones facilitan ilustrar los atributos de brillo y contraste de una imagen, así como comprimir su contenido de ruido, agudizar cada fragmento, entre otras opciones.

La imagen posee componentes de frecuencia que tienen variaciones de bajas o altas frecuencias. Dominan transiciones rápidas de brillo, mientras que las transiciones de brillo que apenas cambian representan bajas frecuencias. Las altas frecuencias en una imagen aparecen cuando están presentes bordes quebrados, como la transición del blanco al negro en dos áreas adyacentes. (Felipe & Suárez, 2015)

El proceso de filtrado en una imagen (f) se fundamenta en administrar una modificación (T) para así conseguir una actual imagen (g) de manera que algunos atributos o características sean recalcadas o reducidas:

$$g(x, y) = T[f(x, y]$$

1.3.11.2. Técnicas de filtrado

El filtrado es un método para mejorar o alterar una imagen (Felipe & Suárez, 2015) existen principalmente dos tipos de filtrado son: Filtrado espacial, Filtrado de frecuencia.

1.3.11.2. Filtro espacial

Las operaciones de procesamiento digital de imágenes son llamadas operaciones de filtrado espacial o filtros de dominio del espacio. En el proceso de filtrado espacial el valor del píxel ya procesado para el píxel evidente necesita de sí mismo y de los píxeles vecinos de modo que el filtrado espacial es un proceso vecino, en donde el valor de cualquier píxel en específico en la salida de imagen se calcula a través del uso de determinados algoritmos con los valores de los píxeles cercanos del debido píxel de entrada, la cercanía de un píxel se determina por un grupo de píxeles circundantes en vínculo con ese píxel. (Felipe & Suárez, 2015).). Algunos tipos de filtrado espacial se discuten a continuación.

1.3.11.3. Filtro Lineal Convolución

El producto del procedimiento de filtrado lineal es la sumatoria de productos máscara con los píxeles correspondientes de manera exacta debajo de la máscara. (Ojeda, 2015)

El filtro lineal se puede expresar por la siguiente ecuación:

$$\begin{aligned} I(x, y) = {} & [M(-1,1) * I(X-1, Y+1)] + [M(0,1) * I(x, y+1)] \\ & + [M(1,1) * I(x+1, y+1)] + [M(-1,0) * I(x-1, y)] \\ & + [M(0,0) * I(x, y)] + M(1,0) * I(x+1, y)] + [M(-1,-1) * I(x-1, y \\ & - 1)] + [M(0,-1) * I(x, y-1)] + [M(1,-1) * I(x+1, y-1)]. \end{aligned}$$

El coeficiente de máscara M (0, 0) se solapa con el valor de píxel de imagen I (x, y), que simboliza que el centro máscara se ubica en (x, y) en el momento que el cálculo de la suma de productos ocurre, no obstante para una máscara de dimensión p x q , q y p son números impares y simboliza a p = 2m + 1, q = 2n +1 en el cual m y n no son negativos enteros, en donde el filtrado lineal se realiza en una imagen (I) con un tamaño p x q, pero con una máscara de tamaño con filtro p x q se torna dada por la siguiente ecuación (Ojeda, 2015):

$$LF(x,y) = \sum_{a=-m}^{m} \sum_{b=-n}^{n} M(a,b) * I(x+a, y+b).$$

1.3.12. Aprendizaje supervisado

El aprendizaje supervisado a partir de datos de entrenamiento deduce una función, su meta es aprender fórmulas generales que mapean entradas y salidas. (Sancho, 2018, págs. 19-33)

1.3.13. Algoritmo de aprendizaje supervisado: K-NN

El principal propósito de este filtro es mejorar el detalle fino en una imagen o destacar el detalle borroso , el afilado se puede desarrollar usando derivadas espaciales las cuales pueden ser aplicadas en áreas de regiones planas o nivel gris constante, en el paso y al final de discontinuidades o discontinuidades de rampa, debido a los desajustes como las líneas, bordes y puntos de ruido, no obstante las derivadas espaciales parciales de primer orden de una imagen digital I (x, y), (Gampala, Gurugubelli, & Hari, 2018, págs. 91-100) , se pueden expresar utilizando la siguiente ecuación:

$$\frac{\partial I}{\partial x} = I(x+1, y) - I(x, y) \;\; y \;\; \frac{\partial I}{\partial x} = I(x, y+1) - I(x, y)$$

Primer orden derivado parcial debe ser: (1) cero en regiones planas, (2) diferente de cero en las discontinuidades de paso y de rampa de nivel de gris y (3) distintas de cero a lo largo de rampas, las derivadas parciales espaciales de segundo orden de la imagen digital I (x, y) se pueden expresar mediante la siguiente ecuación:

$$\frac{\partial^2 I}{\partial x^2} = I(x+1,y) + I(x-1,y) - 2I(x,y) \frac{\partial^2 I}{\partial y^2} = I(x,y+1) + I(x,y-1) - 2I(x,y)$$

Segunda derivada parcial orden debe ser: cero en regiones planas, diferente de cero en el paso y discontinuidades gris nivel de rampa, cero a lo largo de rampas de pendiente constante, la derivada de primer orden es diferente de cero a lo largo de toda la discontinuidad o rampa, no obstante la derivada de segundo orden es diferente de cero únicamente en el paso y gris discontinuidades nivel de rampa, de igual manera la derivada de primer orden se emplea para hacer el borde de espesor y un derivado de segundo orden se usa para desarrollar los detalles finos relevantes como bordes y líneas finas, incluyendo ruido

1.3.14. Segmentación

Es un procedimiento donde la imagen es separada en regiones o componentes las cuales pueden pertenecer a objetos o segmento de objetos, así como se ocupa de determinar si cada uno de los píxeles de la imagen corresponde al objeto de interés.

Es uno de los pasos de mayor importancia que conducen al análisis de los datos de imagen procesada, su objetivo primordial es fraccionar la imagen en partes que poseen gran correlación con las áreas de la realidad contenida en la imagen, podemos apuntar para la segmentación completa, lo que resulta en un conjunto de regiones disjuntas correspondientes de forma única con los objetos de la imagen de entrada, o para la segmentación parcial, en el que las regiones no se corresponden directamente con los objetos de imagen. El proceso de segmentación total de una imagen digital R, esto es un conjunto finito de cada región R1, …, RS (Sonka, Hlavac, & Boyle, 1993).

1.3.15. Extractores de Características.

Está basada mediante la identificación y descripción de los objetos presentes en las características de una imagen tales como el nivel de intensidad, color, textura, posición de la estructura geométrica, superficie, entre otros, también depende de la aplicación concreta que se la da. hay varias formas para obtener las características, pero hasta la actualidad no hay un método que sea el método universal. (Viera Maza, 2017).

1.3.16. Filtro Gaussiano

El filtro gaussiano tiene facilidad de suavizar la imagen quitando el ruido: el filtro se puede aplicar en ambas dimensiones de una imagen de forma independiente. Se define un filtro unidimensional que opera verticalmente y otro que opera horizontalmente, y aplica ambos; esto produce el mismo efecto que un solo filtro aplicado en dos dimensiones.

1.3.17. Thresholding

Convierte todos los píxeles en negro o blanco y en base a eso busca los contornos de máscara y separa las regiones de una imagen, regiones que corresponden a los objetos que se desean analizar. Esta separación son los cambios de intensidad entre los píxeles de los objetos y píxeles del fondo. Para poder diferenciar entre los píxeles que nos interesan de los que no nos interesan que eventualmente sería rechazado, se hace una comparación de cada uno de los valores de la intensidad de cada píxel con respecto a un umbral, determinado según los diferentes problemas a enfrentar.

1.3.18. Operaciones morfológicas

El objetivo primordial de una innovación morfológica es la extraer las estructuras de tipo geométricas esto en relación a que tipo conjuntos que usa, la forma de uso del conjunto de manera conocida, a esto se le conoce

como estructurante o elemento estructurante. Donde la forma y el tamaño del elemento estructurante se toma a priori, esto es acorde a la morfología a la que interseciona en función de las obtenciones de formas que requiere extraer.

1.3.19. Detección de contornos

La detección de bordes o contornos incluye a varios de los métodos matemáticos que apuntan a reconocer los puntos en una imagen digital donde el brillo cambia bruscamente o tiene interrupciones, donde los puntos en los que el brillo cambia bruscamente en una imagen y se organizan en un conjunto de varios segmentos de líneas y las curvas que son denominados los bordes. Donde el problema de hallar discontinuidades en señales de una dimensión s se conoce como detección de pasos y el problema de hallar discontinuidades en las señales en el tiempo sería la detección de permutaciones. En procesamiento la detección de bordes es una de las mejores herramientas en las áreas de detección y también en la extracción características.

1.3.20. TensorFlow

Es la librería de software con acceso de código abierto para el flujo de la programación y de los datos de esta cuenta con una diversa variedad de tareas. Así también otros autores mencionan que es una librería simbólica matemática que se utiliza en aprendizaje automático como en las redes neuronales. Es de gran utilidad para el campo de investigación y para la productividad en Google donde es muy usado.

El aprendizaje automático con TensorFlow brinda a los lectores una base sólida en conceptos de adiestramiento automático, además de la destreza práctica en la codificación de TensorFlow con Python. (Nishant, 2018)

1.3.20.1. Modelo de ejecución TensorFlow

El presente modelo usa un solo flujo de datos para simbolizar el cálculo total y la etapa de un algoritmo en el de aprendizaje

automático, las operaciones matemáticas individuales están incluidas, Las reglas los parámetros de datos en procesamiento. (Nishant, 2018)

Dataflow hace que la comunicación entre subcomputaciones sea explícita y por ende, permite la ejecución de cálculos independientes en paralelo y la partición de los cálculos entre múltiples dispositivos distribuidos. Dataflow hace que la comunicación entre subcomputaciones sea explícita y, por lo tanto, ayuda a la ejecución de cálculos independientes en paralelo y la partición de los cálculos entre múltiples dispositivos distribuidos. La diferencia de los sistemas de flujo de datos entre el Dataflow y TensorFlow por lotes es debido a los siguientes parámetros:

- El patrón acepta diferentes acciones de manera paralela en sub gráficos que se encuentran superpuestos al gráfico real.
- Pueden tener un variado estado los vértices particulares, los cuales se pueden compartir entre una variedad de acciones del esquema.

1.3.20.3. Elementos del gráfico de flujo de datos

Los vértices en un esquema de TensorFlow, hacen referencia a una sección atómica de cálculo y los bordes describen la salida, la entrada de un vértice. Se refiere al cálculo en los vértices como operaciones y a los valores que fluyen a lo largo de los bordes como tensores, porque TensorFlow está diseñado para cálculos matemáticos y utiliza tensores (o matrices multidimensionales) para representar todos los datos en esos cálculos.

1.3.21. Keras

Denominada como aprendizaje profundo escrito en Python, que tiene su aplicabilidad sobre la plataforma de aprendizaje automático TensorFlow. tiene como enfoque en permitir una experimentación a la brevedad posible para eso fue desarrollado. Ser capaz de pasar de la idea al resultado lo más rápido posible es clave para hacer una buena investigación.

1.3.22. Python

Lenguaje de programación Python es de alto nivel el cual tiene como propósito general. EL diseño de Python y su filosofía la cual enfatiza que el código tenga legibilidad con sus notables espacios en blanco. Esta arquitectura del lenguaje y el enfoque se encuentran orientados a objetos donde su meta es ayudar a desarrolladores a escribir código lógico claro para proyectos a pequeña y gran escala, es muy utilizado para inteligencia artificial y procesamiento de imágenes.

1.3.23. OpenCV

Es una biblioteca que tiene 2500 algoritmos mejorados a más, donde se incluye el conjunto algoritmos de aprendizaje automático y visión artificial. Los algoritmos se usan para la detección y reconocimiento de caras, rastrear objetos en pleno movimiento, identificar objetos, clasificación de videos, los movimientos en pistas de video, fusionar, encontrar las similitudes en imágenes en una base de datos, también en las imágenes tomadas con flash puedes descartar los ojos rojos, seguir el movimiento ocular, reconocimiento de panorama y poner marcadores para revestirlo de realidad aumentada. Mas de 47 mil personas tiene OpenCV en su comunidad donde muchos son desarrolladores los cuales generan descargas superiores a 18 millones. Es utilizada en gran parte por las entidades privadas, grupos de investigación y por los organismos estatales (OpenCV, 2019).

1.4. Formulación del Problema.

¿Cómo desarrollar un método para la identificación de las personas utilizando las huellas palmarias deterioradas del personal de campo?

1.5. Hipótesis.

Desarrollo de un método de identificación basado en las palmas de las manos con deterioro.

1.6. Justificación e importancia del estudio.

1.6.1. Justificación Tecnológica.

Se fomenta y acoge el uso de las nuevas tecnologías en la empresa. Se utilizaron tecnologías de software libres adaptables para el desarrollo del método.

1.6.2. Justificación Operativa.

El nuevo método a desarrollar permitirá identificación biométrica de los usuarios con problemas en las manos como mutilaciones suciedad o daños en las huellas dactilares.

1.6.3. Justificación Social.

La implementación del método no sólo implica relevancia tecnológica sino también humana debido a que ayudará a la identificación de los usuarios de campo los cuales cuentan con las manos afectadas por el trabajo duro al cual están expuestos.

1.7. Objetivos.

1.7.1. Objetivo general.

Desarrollar un método de identificación basado en la huella palmaria a través del procesamiento de imágenes digitales.

1.7.2. Objetivos específicos.

a) Preparar el Dataset de las imágenes digitales de palmas de manos.
b) Selección de métodos de clasificación en literatura.
c) Diseñar un nuevo método en base a las técnicas seleccionadas.
d) Codificar en lenguaje de programación el método.
e) Probar y evaluar el resultado del nuevo método planteado.

II. MATERIAL Y MÉTODO

2.1. Tipo y Diseño de Investigación.

2.1.1. Tipo de investigación

La presente investigación corresponde al tipo de investigación **cuantitativa**, porque tiene como objetivo realizar mediciones de indicadores.

2.1.2. Diseño de la investigación

El tipo de investigación empleado es de tipo cuasiexperimental, la investigación consiste en las pruebas de la variable sin ningún tipo de selección que pueda cambiar.

2.2. Población y muestra.

2.2.1. Población

La población está formada por 26 redes neuronales convolucionales según la revisión de la literatura hasta el año 2020 estas son Xception, DenseNet201, VGG16, DenseNet169, VGG19, NASNetLarge, ResNet50, , MobileNetV2, ResNet101, ResNet152, MobileNet, InceptionResNetV2, ResNet50V2, InceptionV3, ResNet101V2, DenseNet121, ResNet152V2, GoogleNet, EfficientNetB0, EfficientNetB1, EfficientNetB2, EfficientNetB3, EfficientNetB4, EfficientNetB5, EfficientNetB6, EfficientNetB7.

2.2.2. Muestra

En esta investigación se utilizó el muestreo no probabilístico y estadística finita por conveniencia por las siguientes razones, existen 26 redes y se han seleccionado 6 redes neuronales por el desempeño según la literatura esto corresponde a una muestra estadística, por conveniencia las redes son: VGG16, VG19, ResNet50, MobileNetV2, Xception y DenseNet121.

2.3. Variables, Operacionalización.

2.3.1. Variable Dependiente:

Las redes neuronales convolucionales.

2.3.2. Variable Independiente:

Identificación de las personas.

Tabla 1.

Operacionalización de Variables

VARIABLE INDEPENDIENTE	INDICADOR	FÓRMULA	TÉCNICA
REDES NEURONALES	Consumo de CPU	$Cc = \sum_{j}^{n} cc_j/n$	Instrumento Electrónico.
	Consumo de Memoria RAM	$U\ M = TM = TI - TF$	
	Tiempo de respuesta	$TTR = TI - TF$	
VARIABLE DEPENDIENTE	INDICADOR	FÓRMULA	TÉCNICA
IDENTIFICACIÓN DE LAS PERSONAS	Precisión	Precisión = VP / FP + VP	Instrumento Electrónico
	Exactitud	Exactitud= VP+VN/VN+FP+FN+VP	Instrumento Electrónico
	Recall	R= VP / FN + VP	Instrumento Electrónico.
	Valor-f	$F1 = 2.\frac{precis.recall}{precis + recall}$	Instrumento Electrónico

2.4. Técnicas e instrumentos de recolección de datos, validez y confiabilidad.

2.4.1. Observación

Mediante esta técnica, se pretende observar el comportamiento del método analizando su comportamiento durante su implementación.

2.4.2 Instrumentos

En esta investigación se usó instrumentos mecánicos y electrónicos a través de una matriz de confusión para conocer los errores que tuvo el modelo.

Los datos a recopilar son verdadero positivo verdaderos negativos la matriz de confusión tiene esta forma:

Tabla 2.

Matriz de confusión

		La Predicción	
		Positivos	**Negativos**
Observación	***Positivos***	Verdadero Positivo	Falso Positivo Positivos
	Negativos	Falso Positivo Positivos	Verdadero Positivos

2.5. Procedimientos de análisis de datos.

Para evaluar el método se utilizaron los siguientes indicadores:

2.5.1. Consumo de CPU

El Consumo de CPU es la cantidad de tiempo que se usó la unidad central de procesos en este caso para procesar la ejecución de la red convolucional para el entrenamiento y predicción del método propuesto para calcular el consumo de CPU usamos la fórmula siguiente:

$$Cc = \sum_{j}^{n} \quad cc_j/n$$

Donde:

El consumo del CPU Cc es la sumatoria del consumo todas las ejecuciones cc_j entre el número de ejecuciones $/n$

2.5.2. Consumo de memoria RAM

La ejecución de un programa en un computador hace uso de memoria RAM donde almacena la información temporal de las ejecuciones que se realizan, en esta investigación se utiliza redes neuronales convolucionales que ocupan memoria para las variables que utilizan, una red neuronal convolucional tiene capas entre ellas capas polling, capas de reducción, capas de convolución y todos estos datos son almacenados en la RAM, las redes neuronales tienen mucho consumo de memorias RAM por lo tanto es un indicador para la medición.

$$Utilización\ de\ Memoria = TM = TI - TF$$

Donde:

$$TM = Total\ de\ MemoriaUM = Uso\ de\ Memoria$$

TM es el total de memoria

2.5.3. Tiempo de respuesta

Tiempo que toma ejecutarse el clasificador. La fórmula a emplearse es:

$$TR = TI - TF$$

Donde:

$$TI = Tiempo\ de\ InicioTF = Tiempo\ Final$$

TR= Tiempo de respuesta

2.5.4. Precisión

La precisión o en ingles accuracy calculó la cantidad total de predicciones positivas que la red neuronal convolucional a acertado para ello se utilizó la siguiente formula

$$precisión = \frac{VP}{VP + FP}$$

Donde:

VP = Verdaderos positivos o de predicciones correctas.

FP = son la cantidad de falsos positivo pero el valor es negativo

2.5.5. Exactitud

Es la cantidad Total de predicciones acertadas de las redes neuronales convolucionales en base al número de predicciones realizadas

$$exactitud = \frac{VN + VP}{VP + FP + FN + VP}$$

Donde:

VN: Verdaderos negativos o predicciones correctas.

VP: Verdaderos positivos o de predicciones correctas.

FP: Falsos positivo pero el valor es negativo.

FN: La predicción es negativa cuando en realidad es positivo.

2.5.6. Recall

Con la exhaustividad o Recall se pudo calcular la cantidad final de predicciones de las manos como la tasa de verdaderos positivos (TPR) que realizó el método propuesto, para ellos se aplicó la fórmula siguiente:

$$Recall = \frac{VP}{FN+VP}$$

Donde:

Variable	Especialización
VP	Verdadero Positivo
FN	Falso Positivo

2.5.7. Valor-F

EL valor-F o también conocido en la literatura como el F score o F1, es el indicador que medirá nuestro test, este valor se calcula a partir de nuestra precisión y nuestra exhaustividad, donde la precisión es la cantidad resultados positivos o en esta investigación el resultado de manos que fueron correctamente identificadas y dividido por el número positivos, se incluyen los que fueron identificados correctamente y la cantidad de los resultados positivos que se identificado correctamente esto es dividido por la cantidad de muestras debieron ser reconocidas como positivas.

Para el desarrollo de esta tesis se usará la siguiente fórmula basada en valores de la precisión y Recall.

$$F1 = 2.\frac{precis.recall}{precis + recall}$$

Donde:

Variable	Especificación
Pres	precisión
Racall	Recall

2.6. Criterios éticos

Se enmarca en tres criterios éticos; confidencialidad, derecho de autor y búsqueda de bien:

2.6.1. Confidencialidad

La investigación cumplió el código ético, la seguridad y también con protección de las identidades de los participantes, teniendo como base el reconocimiento de respeto a la autonomía y a la dignidad del ser humano

2.6.2. Derechos de autor

Para el desarrollo de esta investigación se respetó el buen uso de la información citando a sus respectivos autores y referenciándolos de manera adecuada.

.

2.6.3. Búsqueda del bien

En esta investigación se buscó el bien donde la obligación fue los máximos beneficios posibles y no tener posibilidad de daños e injusticias para que la investigación esté bien concebida, y se buscó en todo momento ser competentes para llevar cabo la investigación y garantizar el bienestar de los participantes.

.

2.7. Criterios de Rigor Científico.

2.7.1. Originalidad

Esta investigación se desarrolló en base al trabajo de investigación y está citado por fuentes bibliográficas, para reflejar la ausencia del plagio.

2.7.2. Confiabilidad:

La investigación se aseguró la confiabilidad porque está basada en la estadística de la matriz de confusión y la estadística que establece la exactitud.

2.7.3. Validez:

La validez en este caso fue de tipo lógica donde la definición de las dimensiones y la operacionalización de las variables están descritas de manera clara y precisa. Donde se evalúan las dimensiones mediante los

indicadores que se establecieron y las técnicas definidas para la recopilación de datos, el proyecto de investigación evaluó y analizó.

2.7.4. Consistencia

La investigación fue de carácter científico. El análisis aplicado a los datos está hecho con suma profesionalidad usando técnicas e información de ingeniería e investigación para mantener la coherencia y eficacia de los datos.

III. RESULTADOS

3.1. Resultados en Tablas y Figuras

A continuación, se presentan los resultados logrados por las redes convolucionales en sus dos etapas de entrenamiento y testing, para realizar las mediciones de estos indicadores en etapa de entrenamiento, se hizo uso de la herramienta administrador de procesos de Windows que permiten visualizar el porcentaje utilizado por el CPU y la memoria RAM en tiempo real en el momento de cada uno de los experimentos, y en etapa de testing se hizo uso de la librería de Python Sklern que permiten hacer mediciones exactas del Tiempo de respuesta, Precisión, Exactitud, Recall, Valor-f se utilizaron estas herramientas porque según la literatura revisada son los más usados y adecuados para medir los resultados de las redes neuronales convolucionales, se entrenaron y evaluaron 6 redes neuronales seleccionadas que son la VGG16, MobileNetV2, VGG19, Xception, ResNet50, DenseNet121, es decir el número de mediciones de indicadores que se realizó, fue de 1 vez por cada experimento o entrenamiento de cada una de las redes neuronales convolucionales.

Las redes convolucionales en la etapa de entrenamiento crearon modelos en ese momento fueron evaluados como es el caso de la medición de consumo de CPU, este indicador permitió conocer cuál es el consumo en porcentaje que utiliza cada red neuronal. La ejecución de cada una de las redes convolucionales está relacionada con el uso del CPU, por los entrenamientos, ejecuciones que se realizaron pues estos procesos hicieron

un uso del CPU, para esta investigación es importante conocer este indicador puesto que se propone el método como una posible solución a un ambiente de producción donde se requerirá el mejor rendimiento del sistema. Para ello se realizó 1 número de experimentos y 1 número de veces de medición de consumo de CPU por la ejecución de cada red neuronal convolucional como se observa en la tabla 3.

Tabla 3

Número de experimentos y mediciones para Consumo de CPU

N°	Red Neuronal Convolucional	Experimentos	Mediciones
1	VGG16	1	1
2	MobileNetV2	1	1
3	VGG19	1	1
4	Xception	1	1
5	ResNet50	1	1
6	DenseNet121	1	1

Fuente: Elaboración Propia.

En el siguiente gráfico se aprecia los resultados obtenidos por cada red convolucional donde ResNet50 obtuvo un menor gasto de CPU con respecto a las otras redes neuronales, como de un 3% menos que la VGG19, de 4 % menos de la MobileNetV2, de 6 % menos de DenseNet121, de 7% menos de Xception, 8% menos de la VGG16.

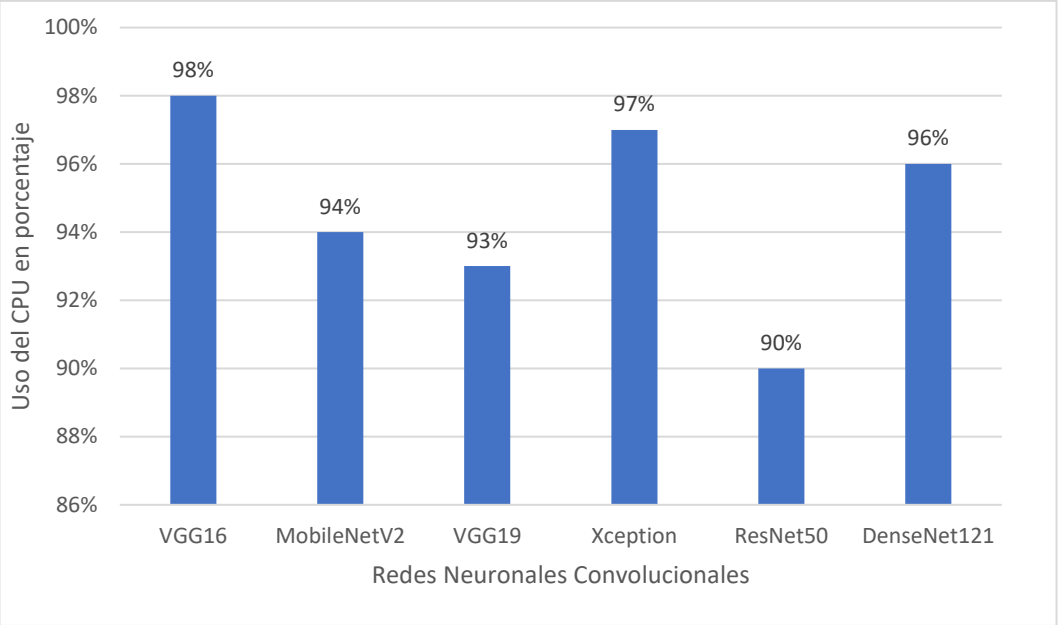

Figura 7 Consumo de CPU en porcentaje

Con esto se pudo conocer que Resnet50 realizo un menor consumo del CPU.

En la etapa de entrenamiento también se realizó la medición del Consumo de memoria RAM pues la ejecución de cada una de las redes convolucionales hace uso de memoria RAM donde almacena la información temporal de las ejecuciones que se realizan, las redes e neuronales tienen mucho consumo de memorias RAM por lo tanto para esta investigación es importante conocer este indicador puesto que se propone el método como una posible solución a un ambiente de producción donde se requerirá el óptimo rendimiento por parte del sistema. Para la medición se realizó 1 número de experimentos y 1 número de veces de medición de consumo de Memoria RAM por la ejecución de cada red neuronal convolucional como se observa en la tabla 4.

Tabla 4

Número de experimentos y medición para Consumo de Memoria RAM.

N°	Red Neuronal Convolucional	Experimentos	Mediciones
1	VGG16	1	1
2	MobileNetV2	1	1
3	VGG19	1	1
4	Xception	1	1
5	ResNet50	1	1
6	DenseNet121	1	1

Fuente: Elaboración Propia.

En el siguiente gráfico se aprecia los resultados obtenidos por cada red convolucional donde ResNet50 y MobileNet obtuvieron 13 porciento, se aprecia que es un menor consumo de Memoria RAM con respecto a las otras redes neuronales, como de menos 1% de la VGG19, de 2% menos de la red VGG16, de 6 % menos de DenseNet121, de 7% menos de Xception.

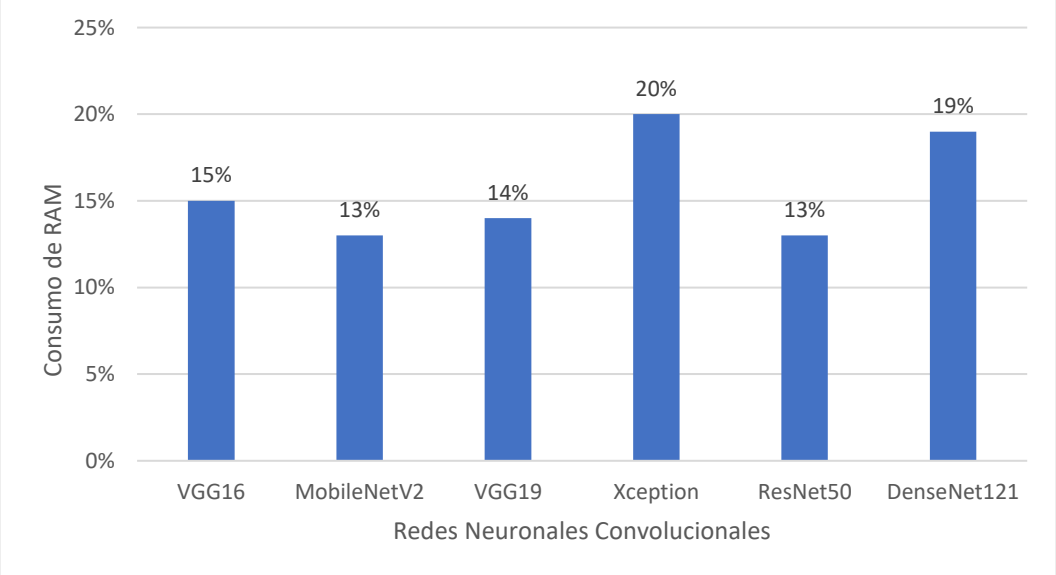

Figura 8. Consumo de RAM en porcentaje.

Fuente: Elaboración Propia.

Se pudo conocer que Resnet50 y MobileNet consumen menos RAM que otras redes.

En la etapa de entrenamiento también se tomó en cuenta la medición del tiempo de respuesta que tardó el sistema en milisegundos, la etapa de entrenamiento de las huellas de palmas de las manos de 100 personas. Es importante conocer este recurso computacional, pues en un ambiente de producción de control de acceso de gran volumen podría fallar en su desempeño.

La medición del tiempo de respuesta de esta investigación fue a través de un contador de tiempo, que se implementó mediante el script de código Python en la etapa de e entrenamiento de cada red neuronal.

Para la medición se realizó 1 número de experimentos y 1 número de veces de medición de consumo de tiempo de respuesta por cada ejecución de cada red neuronal convolucional como se observa en la tabla 5.

Tabla 5.

Número de experimentos y mediciones para Tiempo de Respuesta

N°	Red Neuronal Convolucional	Experimentos	Mediciones
1	VGG16	1	1
2	MobileNetV2	1	1
3	VGG19	1	1
4	Xception	1	1
5	ResNet50	1	1
6	DenseNet121	1	1

Fuente: Elaboración Propia.

En el siguiente gráfico se muestran el resultado de cantidad de tiempo de respuesta de la ejecución de cada una de las redes neuronales donde VGG16 obtuvo 2616321ms, MobileNetV2 obtuvo 2316234 ms, VGG19 obtuvo

2516453 ms, Xception 2816345 ms, ResNet50 obtuvo 2716442 ms y DenseNet121 obtuvo 2916442.

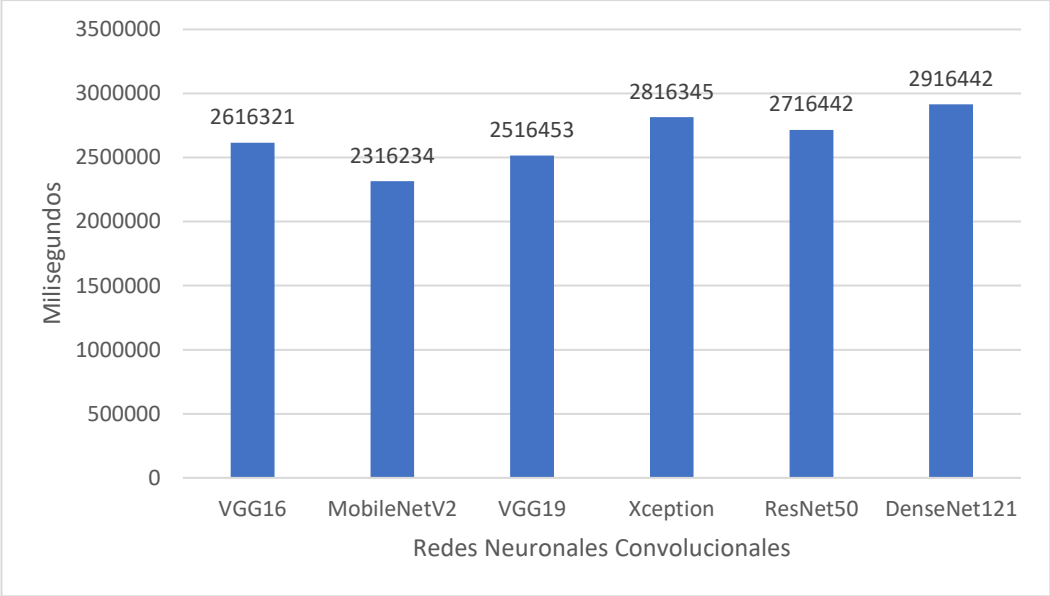

Figura 9. Tiempo de respuesta en Milisegundos de las redes neuronales. Fuente: Elaboración Propia.

En el gráfico se muestran el tiempo promedio de respuesta de cada red. Donde la red neuronal MOBILNETv2 el tiempo respuesta más bajo, es decir que el proceso de entrenamiento es más rápido que el resto de redes neuronales.

Para la etapa de entrenamiento de aprendizaje de las imágenes digitales de huellas de las palmas de manos se realizó la medición del porcentaje la cantidad de precisión de manos reconocidas.

La precisión es importante para conocer el desempeño del método de aprendizaje de huellas palmarias.

Para la medición se realizó 1 número de experimentos y 1 número de veces de medición de la precisión por cada ejecución de cada red neuronal convolucional como se puede apreciar en la siguiente tabla.

Tabla 6.

Número de experimentos y mediciones para la precisión

N°	Red Neuronal Convolucional	experimentos	Mediciones
1	VGG16	1	1
2	MobileNetV2	1	1
3	VGG19	1	1
4	Xception	1	1
5	ResNet50	1	1
6	DenseNet121	1	1

Fuente: Elaboración Propia

En la tabla se pudo apreciar que RestNet50 con 98.5% es la mejor red neuronal en precisión a 1.5% mayor de DenseNet121 siendo estas 2 redes neuronales las que obtuvieron mejor puntaje.

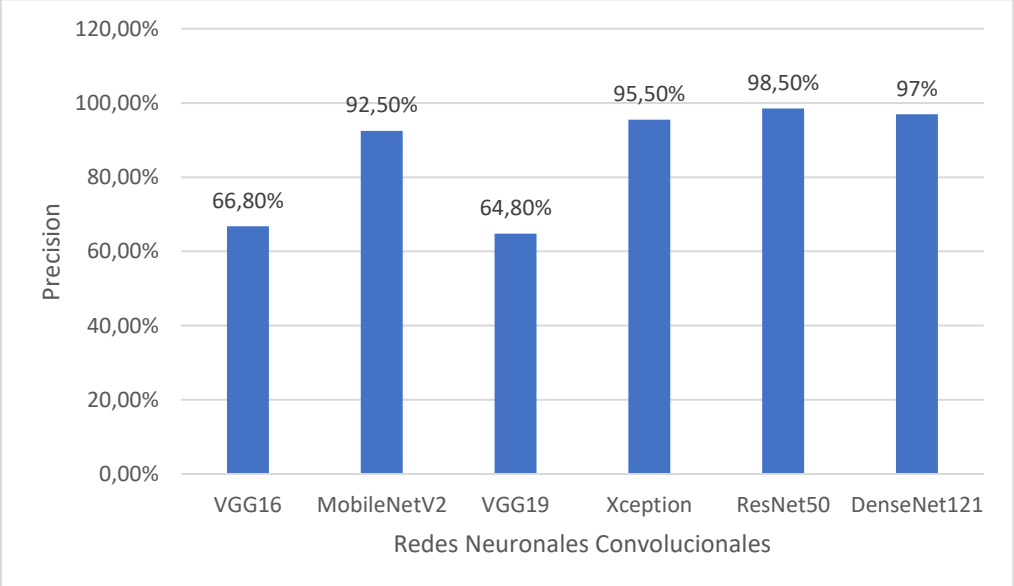

Figura 10. Tiempo de respuesta en Milisegundos.

Fuente: Elaboración Propia

También se puede apreciar en el gráfico que el porcentaje de precisión menor es el de la red neuronal VGG19, es decir que el proceso de reconocimiento de manos con este método no es tan efectivo ni preciso. De esta manera se demostró que la red neuronal ResNet50 fue la que mejor precisión obtuvo en el reconocimiento de manos.

El uso de redes convolucionales para el reconocimiento está relacionado con la exactitud porque permite medir el porcentaje de clases que el modelo de esta investigación ha acertado en el reconocimiento de manos y esto se hizo por cada red.

Los resultados se obtuvieron de observación de números de aciertos al ejecutar cada una de las 6 redes

Para la medición de la exactitud se realizó 1 número de experimentos y 1 número de veces de medición de la exactitud por cada ejecución de cada red neuronal convolucional como se puede apreciar en la siguiente tabla.

Tabla 7

Número de experimentos y mediciones para la exactitud.

N°	Red Neuronal Convolucional	experimentos	Mediciones
1	VGG16	1	1
2	MobileNetV2	1	1
3	VGG19	1	1
4	Xception	1	1
5	ResNet50	1	1
6	DenseNet121	1	1

Fuente: Elaboración Propia

Los datos mostraron que la red neuronal convolucional ResNet50 obtuvo el mejor resultado con respecto a las otras redes neuronales con más de un

1 % que la DenseNet, de 2 % de la Xception, de 4 % de MobileNetV2, de 24% de VGG16, 8% de la VGG16. Con esto pudimos conocer que Resnet50 realizo un mayor número de aciertos.

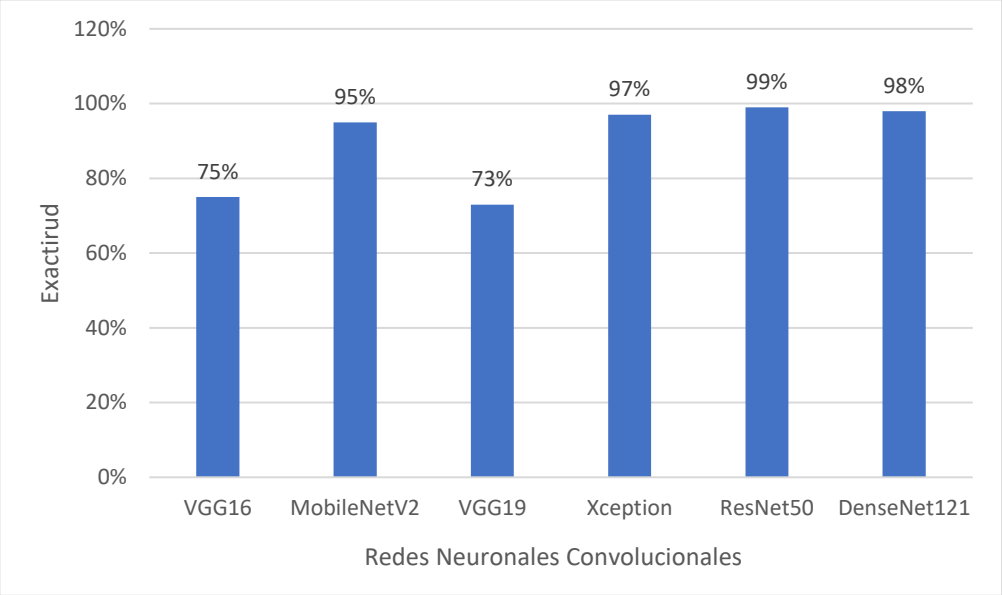

Figura 11. Resultado de la exactitud en porcentaje de cada red neuronal

Fuente: Elaboración Propia.

La métrica de Recall también fue medida en la etapa de trainig y está relacionada a esta investigación pues nos brindó información sobre la cantidad de clases o manos que el modelo fue capaz de identificar.

Para calcular el Recall o la exhaustividad promediamos la cantidad de clases que identificó.Para la medición de la Recall se realizó 1 número de experimentos y 1 número de veces de medición como se aprecia en el siguiente tabla.

Tabla 8

Número de experimentos y mediciones para Recall

N°	Red Neuronal Convolucional	Experimentos	Mediciones
1	VGG16	1	1
2	MobileNetV2	1	1
3	VGG19	1	1
4	Xception	1	1
5	ResNet50	1	1
6	DenseNet121	1	1

Fuente: Elaboración Propia

Los datos de la red neuronal ResNet50 obtuvo el mejor resultado con respecto a las otras redes neuronales con más de un 1 % que la DenseNet, de 2 % de la Xception, de 4 % de MobileNetV2, de 24% de VGG16, 8% de la VGG16. Con esto pudimos conocer que Resnet50 realizo un mayor número de aciertos como podemos apreciar en el siguiente gráfico de resultados.

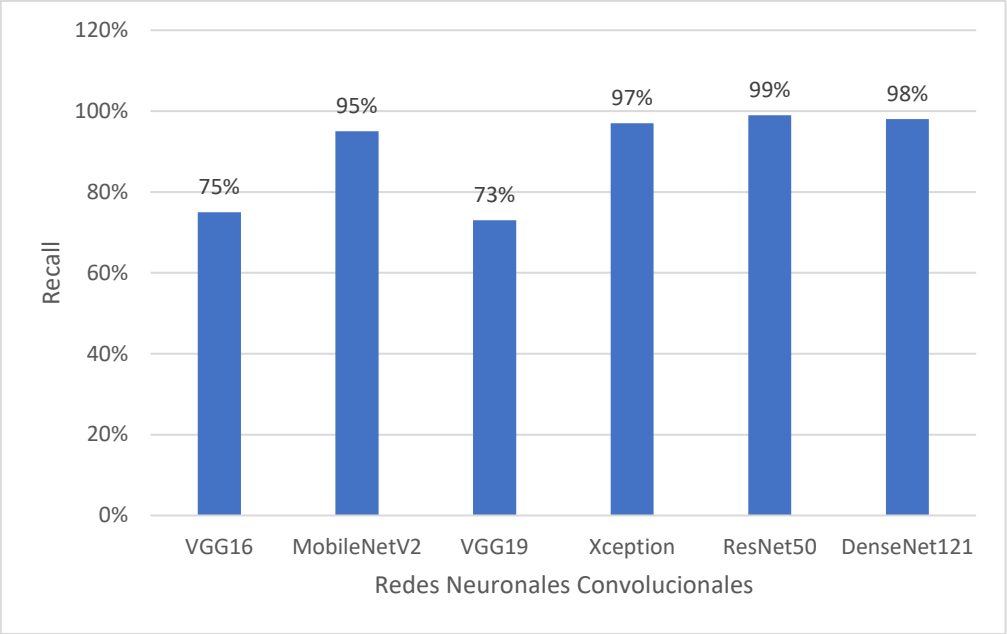

Figura 12. Resultado de Recall en porcentaje de cada red neuronal

Fuente: Elaboración Propia

Valor-F

El valor F en esta investigación está relacionado porque es la medida de precisión que tiene nuestro valor de testing de identificación de las clases o manos y combina los valores de la precisión y de recall. de tal forma que promediamos por las 100 clases o manos y arroja como resultado los siguientes valores.

Tabla 9.

Determinación del Valor-F de cada red neuronal propuesta.

N°	Red Neuronal Convolucional	experimentos	Valor F
1	VGG16	1	0.693
2	MobileNetV2	1	0.933
3	VGG19	1	0.675
4	Xception	1	0.960
5	ResNet50	1	0.987
6	DenseNet121	1	0.973

Fuente: Elaboración Propia

Estos valores al multiplicarlos por 100 para dar como resultado del porcentaje de la precisión que obtuvo el testing como podemos ver a continuación en la siguiente gráfica.

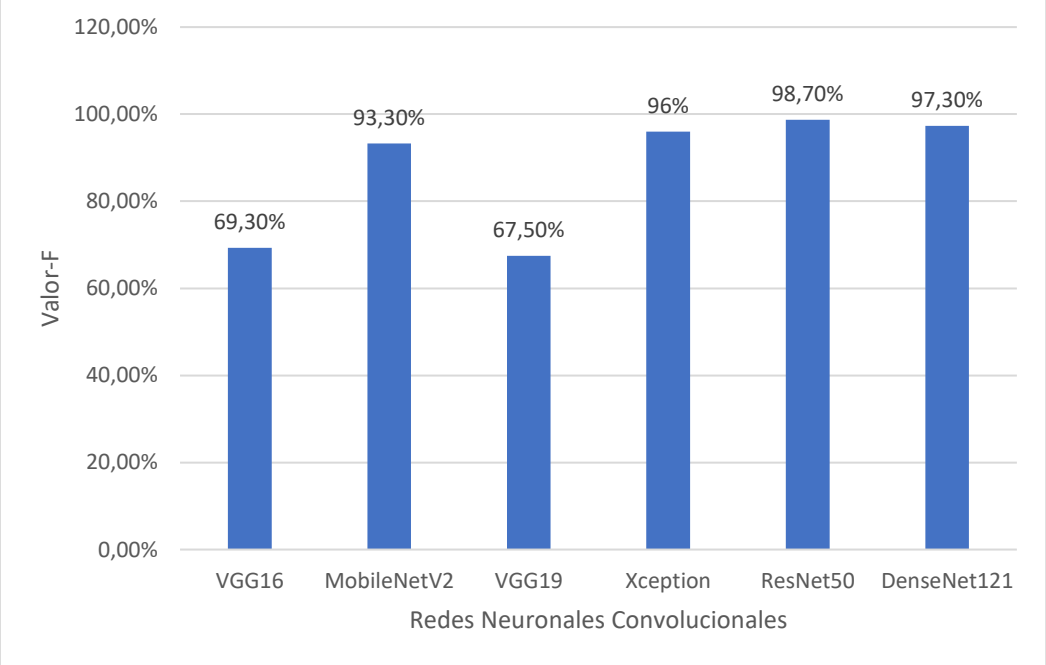

Figura 13. Resultados del Valor F de cada red neuronal.

Fuente: Elaboración Propia

La precisión del testing en la red ResNet50 obtuvo el mejor resultado con respecto a las otras redes neuronales con 98.7 % y a un 1.4 % que la red DenseNet, de 2.7% de la red neuronal Xception, de 5.4 % de MobileNetV2, de 29.4% de VGG16, 31.2% de la VGG19. Con esto pudimos conocer que Resnet50 realizo un mayor número de aciertos.

Finalmente podemos ver los resultados de cada una de las redes en la siguiente tabla.

Tabla 10

comparación resultados de Indicadores de redes neuronales.

Red	CPU	RAM	T. Resp.	Precisión	Exactitud	Recal	Valor-f
VGG16	98%	15%	2616321ms	66,80%	75%	75%	69,30%
MobileNet2	94%	13%	2316234ms	92,50%	95%	95%	93,30%
VGG19	93%	14%	2516453ms	64,80%	73%	73%	67,50%
Xception	97%	20%	2816345ms	95,50%	97%	97%	96.00%
ResNet50	90%	13%	2716442ms	98,50%	99%	99%	98,70%
DensNet121	96%	19%	2916442ms	97%	98%	98%	97,30%

Fuente: Elaboración Propia

En la tabla 10 se pudo apreciar un ranking de los resultados obtenidos de cada uno de los indicadores de las pruebas donde se determinó que red neuronal tiene el mejor rendimiento para la tarea de reconocimiento de manos la cual es red neuronal ResNet50 que alcanzo el 99 % en Exactitud y Recall superando a las demás redes neuronales.

3.2. Discusión de resultados

En la investigación “Disease based on Gravitational Search Optimization Algorithm” tuvo como resultado 98% pero en esta investigación se obtuvo 99 % de precisión pues la calidad de las imágenes usadas en esa investigación es baja mientras aquí las imágenes tienen 333 ppi lo que demuestra que el presente trabajo tiene una mejor muestra. En esta investigación se obtuvo 99 % de exactitud con la red neuronal Resnet50 demostrando la eficacia del metodo, En la investigación “Classification and Grading of Okra-ladies finger using Deep Learning” se obtuvo 99 % de exactitud con la red neuronal Resnet50, sin embargo en esta investigación se obtuvo también 99 % si observamos el trabajo de las imágenes de “Classification and Grading of Okra-ladies finger using Deep Learning” la cual tuvo un dataset de 3200 fotografías tomándole una muestra por clase mientras que en esta investigación se usaron muestras de 200 imágenes para entrenar la red y 100 imágenes para predecir sabiendo que las redes convolucionales convergen mejor mientras más imágenes se tenga. Es por ello que esta investigación a pesar de su baja cantidad de muestras tiene un muy buen resultado. En el papper del ingeniero Mejia Cabrera Heber Ivan y compañeros en “New method for subject identification based on palm print” obtuvieron 100 % de exactitud. Sin embargo, esto se dio por que en el papper New method for subject identification based on palm print se trabajó con 50 personas tomándose una muestra de imagen de mano a cada persona y generaron imágenes rotadas, siendo la misma mano repetida varias veces, mientras que en esta investigación se tomaron 3 muestras de imagen de la palma de mano izquierda y 3 muestras de imagen de la palma de la mano derecha por persona a una cantidad de 100 personas quedando mejor representada la muestra.

3.3. Aporte práctico

3.3.1 Preparar Dataset de imágenes digitales de huellas palmarias

Para el cumplimiento del primer objetivo fue necesario construir un dataset con imágenes digitales de palmas de manos de individuos que usan intensamente las manos, por lo cual tienen afectaciones en las huellas

dactilares y la palma de la mano, para ello se seleccionó a los trabajadores de la construcción civil de las principales constructoras de la ciudad de Lima, debido a la coyuntura de la crisis sanitaria producida por el virus SARS-CoV-2 fue necesario construir un protocolo de bioseguridad antes de ejecutar el protocolo de adquisición de imágenes, donde el primer objetivo es proteger la salud de todos los colaboradores durante la investigación, así como la prevención de la diseminación del virus.

El protocolo de bioseguridad tuvo un conjunto de pasos y normas generales a seguir como son: La forma de utilización de equipos de protección personal. consistente en mascarilla, y careta, el lavado de manos, la desinfección de las manos con alcohol medicinal. El protocolo de bioseguridad completo se muestra en el Anexo 1.

Figura 14. a

Figura 15. b

Figura 17. c

Figura 16. d

Demostración del protocolo de bioseguridad para la adquisición de imágenes a. Explicación de protocolos, b. Protocolo Bioseguridad, c. Protocolo de adquisición, d. Siguiente participante

Se aplicó el protocolo de bioseguridad brindando la información sobre el procedimiento a los participantes, para garantizar la confianza y seguridad durante el proceso de adquisición de impresiones palmarias.

Por otro lado, también se ha requerido de un protocolo de adquisición de imágenes para asegurar que el método de identificación pueda procesarlas de la misma manera. De modo que las imágenes que se obtengan de este protocolo cuenten con un estándar con las mismas condiciones o características en cuanto a luz, distancia y calidad de la imagen digital.

Para el protocolo de adquisición de imágenes se hicieron los 3 pasos, como se ilustra a continuación en la siguiente figura:

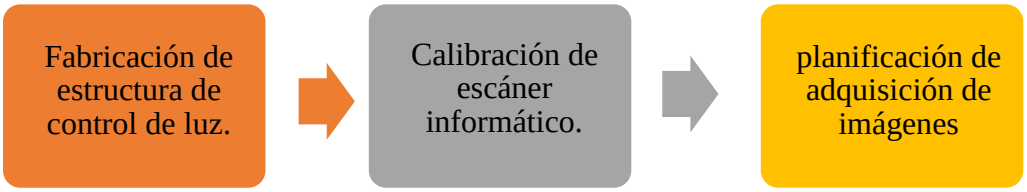

Figura 18. Pasos de Implementación del protocolo de adquisición imágenes

Para la construcción de la estructura se empleó una caja de cartón que tenia de largo, 40 cm de ancho 20 cm y de alto 30 cm estas medidas se establecieron debido a que el dispositivo de adquisición que es el escáner Epson modelo L220 tenía las dimensiones de 38 centímetros de largo 18 centímetros de ancho y 20 centímetros de alto dejando una holgura de 10 centímetros para que el usuario pueda introducir la mano por un agujero ,se usó también 3 piezas de cartulina negra para forrar la caja ni se refleje en el cartón para que la luz exterior no ingrese al dispositivo , de manera que la caja fue forrada por dentro y por fuera con la cartulina quedando completamente negra dejando un pequeño espacio para que el usuario pueda ingresar la mano.

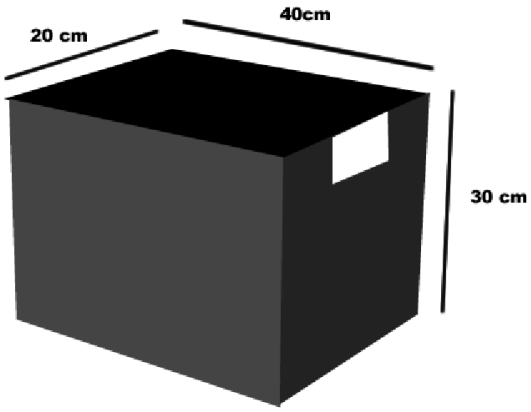

Figura 19. Propuesta de estructura para protocolo adquisición de imágenes

Fuente: Elaboración Propia

En la toma de imágenes se controló la luminosidad para lograr una correcta adquisición de las imágenes. Se utilizó la estructura de cartón negro mostrada en la figura 19 y así conseguir la adquisición de imágenes en cualquier momento del día, posteriormente se ajustó la configuración del escáner a modo de fotografía de oficina para poder obtener una calidad de imagen de alta resolución con los puntos por pulgada (PPP) en 333. En los bordes superiores del escáner se realizaron marcas para que los usuarios pudieran posicionar correctamente las manos en 3 posiciones. Las cuales la primera posición de la mano es puesta al centro de la primera marca la segunda posición girada en 3 grados a la izquierda y 3 grados girada derecha de la posición central todo esto para que el método de aprendizaje logre aprender las características en diversas posiciones. y así el método pueda identificar las diversas posiciones.

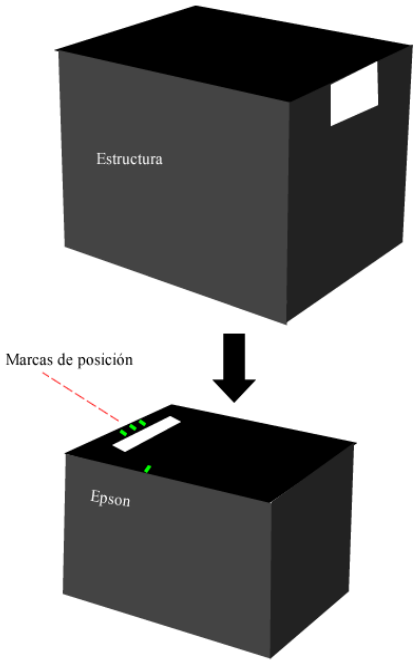

Figura 20. Simulación de Estructura y dispositivo de adquisición, escáner.

Fuente: Elaboración Propia

En la figura 20 se pudo apreciar la forma de operar de la estructura de control de luz con el dispositivo de adquisición de imágenes.

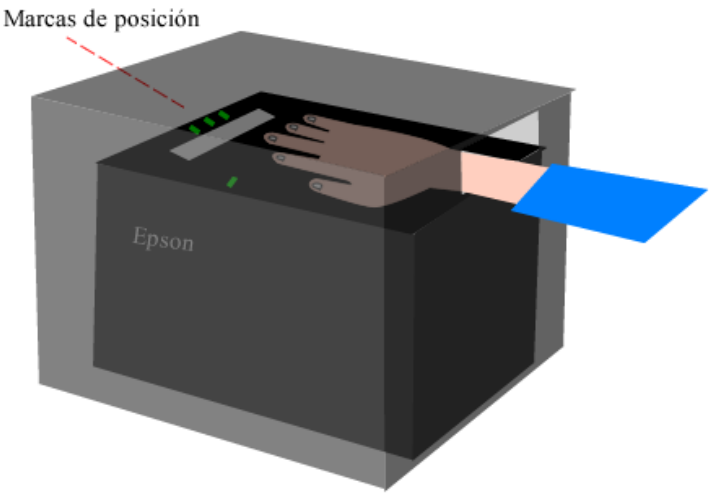

Figura 21. Simulación 3D de estructura y escáner con posición de manos

Fuente: Elaboración Propia

Como se aprecia en la figura 21 la estructura permite el ingreso de la mano de las personas para una correcta postura de las manos en la adquisición de imágenes

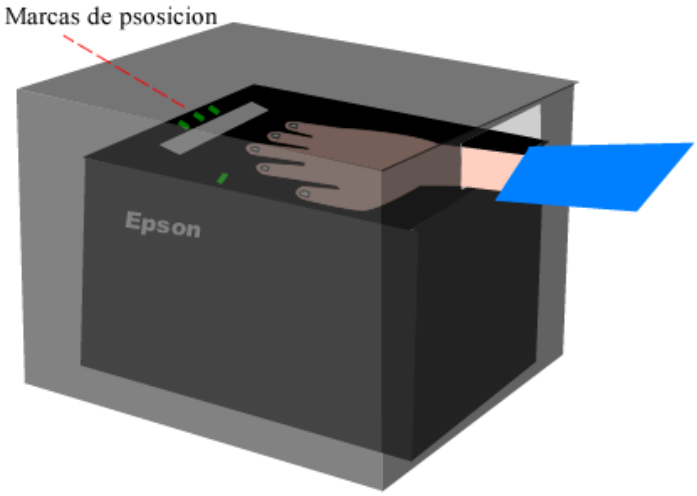

Figura 22. Posición de mano izquierda en marca verde inclinación 3 grados.

Fuente: Elaboración Propia.

En la figura 22 y 23 podemos apreciar 2 de las 3 posiciones correctas para la adquisición de imágenes con la estructura de control de luz.

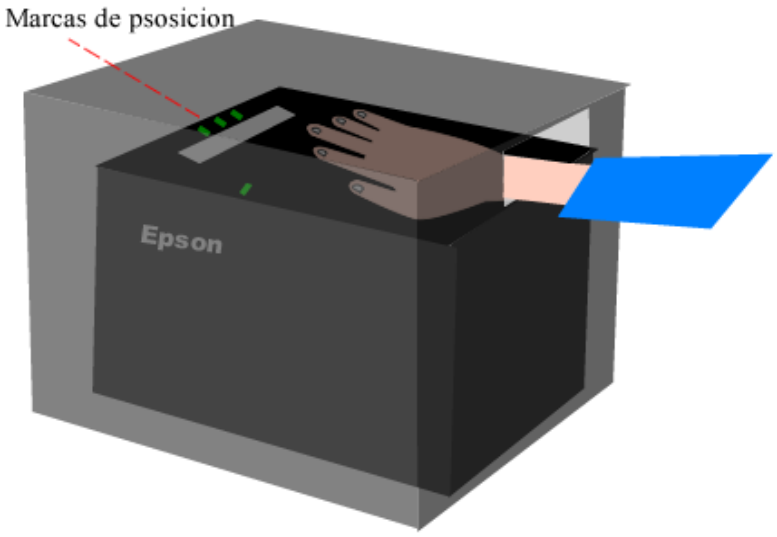

Figura 23. Posicionamiento de mano derecha lado derecho giro de 3 grados

Fuente: Elaboración Propia

Las imágenes adquiridas fueron de 100 personas y quedaron organizadas en distintas carpetas nombradas por números, de tal manera que se pudo diferenciar a los individuos de manera que no se repitieran, visualizándose como se muestra a continuación:

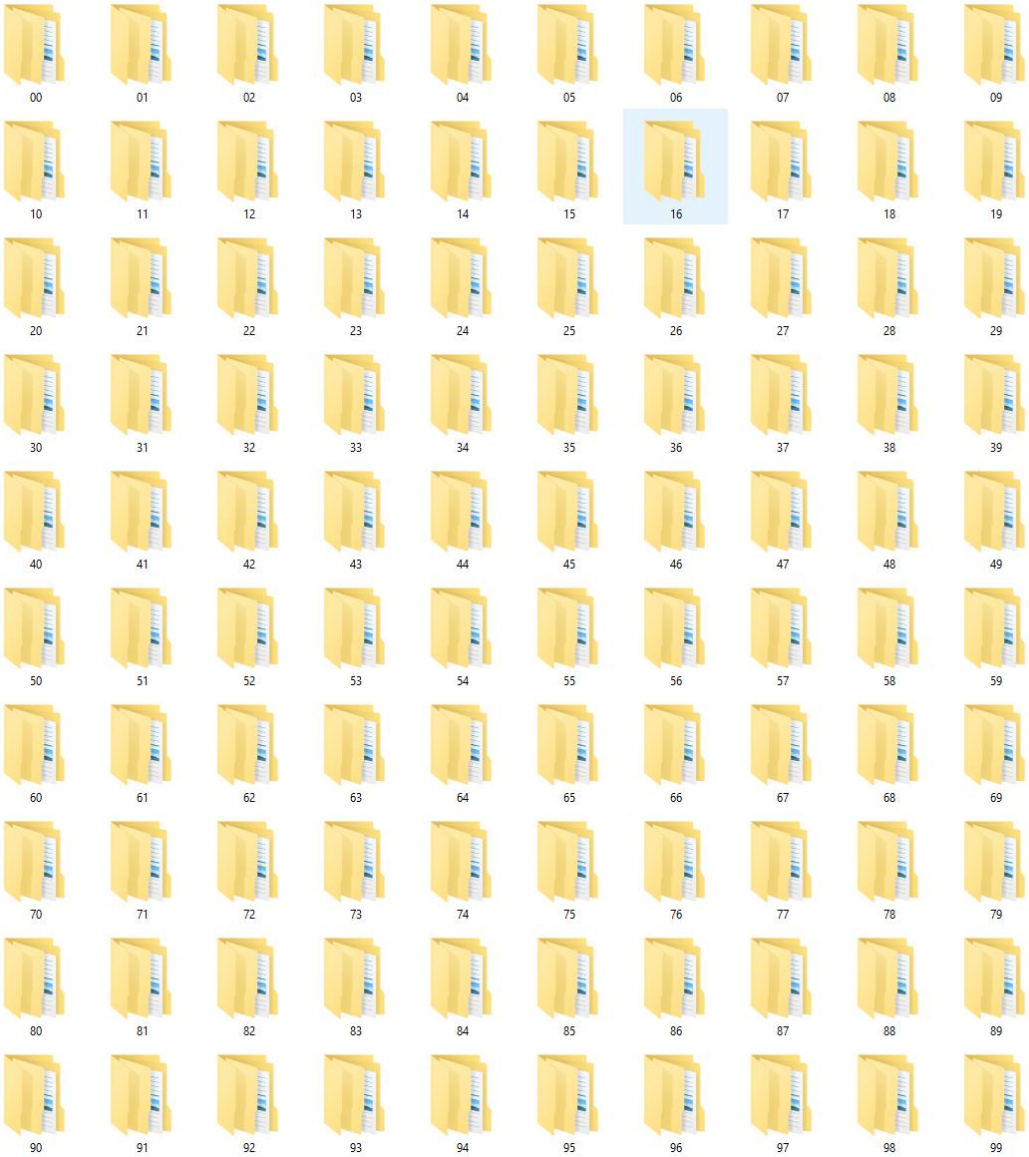

Figura 24 Organización de imágenes adquiridas en carpetas ordenadas.
Fuente: Elaboración Propia

En las siguientes imágenes a b c y d se puede apreciar como quedaron las impresiones palmarias aplicando el protocolo de adquisición de imágenes antes explicado.

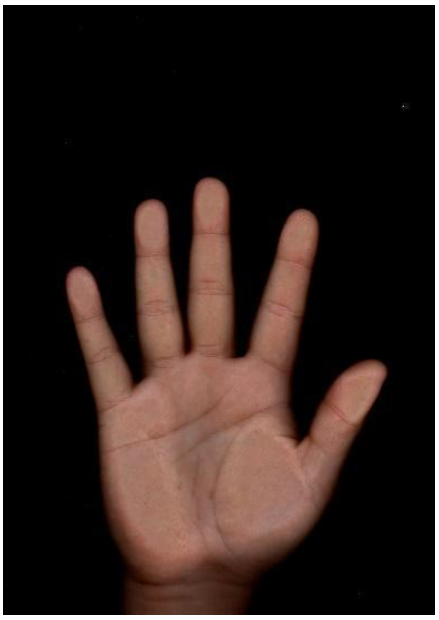

Figura 25. a

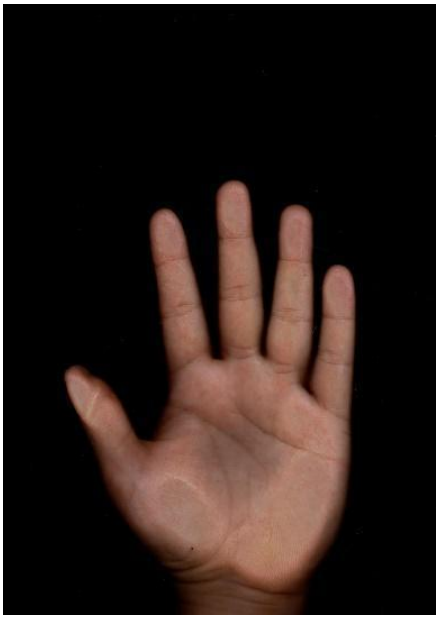

Figura 26. b

Figura 27. d

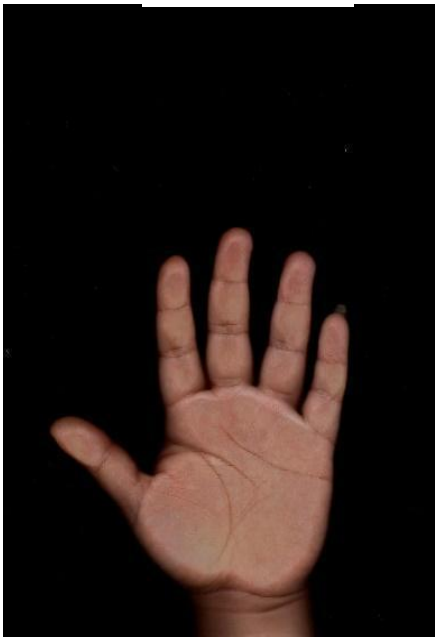

Figura 28.

Imágenes de huellas palmarias reales adquiridas con el protocolo de adquisición

a) Huella palmaria derecha

b) Huella izquierda con afectaciones

c) Huella palmaria izquierda

d) Huella palmaria derecha con uñas

3.3.2 Selección de los métodos de clasificación.

En esta investigación el método de clasificación que se utilizó son las redes convolucionales para ello se hizo un repaso exhaustivo de la literatura que se ha publicado hasta la fecha entre los diversos autores donde se encontró lo siguiente:

Tabla 11

Clasificación de los tipos de redes convolucionales

N	Red Neuronal	Acuracy	Utilización	Autor
1	Xception	99.50%	Para clasificar imágenes de rayos X en tres clases, sustentada en una agrupación de datos de código abierto.	(Mohammad & Abolfazl, 2020)
2	VGG16	100%	Para el desarrollo de un novedoso método para la detección de la huella de la palma	(Mejía, Antón, Flores, Tuesta, & Forero, 2020),
3	VGG19	99.90%	Crea modelos matemáticos, estadísticos o cuantitativos.	(Gamarra, C, & Ríos, 2017)

4	ResNet50	99.0%	Para la clasificación de Okra (cultivo de frutas) para determinar la equidad de precios según la frescura, olor,	(Meenaxi, Meena, Chairtra, & Shantala, 2020)
5	ResNet101	78.43%	Para identificar y clasificar a los pintores por diferentes movimientos artísticos.	(Mehmet, Nurullah, & Tulay, 2019)
6	ResNet152	91.34%	Automatizar la clasificación de imágenes para monitorear la ingesta dietética.	(McAllister, Zheng, Bond, & Moorhead, 2018)
7	ResNet50V2	91.40%	entrenamiento que ayudan a la red a aprender mejor cuando tenemos un conjunto de datos desequilibrado	(Mohammad & Abolfazl, 2020)
8	ResNet101V2	93.80%	Selección de modelo de red neuronal profunda óptimo para el reconocimiento usando imágenes de rayos X.	(Beltrán & Arevalo, 2019)

9	ResNet152V2	99.77%	El estudio propone modelos de aprendizaje profundo eficiente y potente para detectar y clasificar la neumonía.	(Hussein, Ibrahim, & Taha, 2020)
10	InceptionV3	96.64%	Presenta un marco sólido para clasificar las imágenes deportivas en función del entorno y extracción de características	(Ketan, Vikas, Chitransh, & Chaitanya, 2019)
11	InceptionResNet	96.00%	Discriminar entre lesiones benignas y malignas en imágenes de resonancia magnética	(Tomoyuki, Yuka, Jun, & Mori, 2020)
12	MobileNet	94.3%	Para la clasificación y detección automática de células en panal de abejas mediante aprendizaje profundo	(Thiago, y otros, 2020)
13	MobileNetV2	100%	Para el reconocimiento de huellas palmares	(Aurelia, Vincent, & Díaz, 2019)

14	DenseNet121	98%	Se usó en el diagnóstico COVID-19 basada en la optimización.	(Dalia, Aboul, & Hassan, 2020)
15	DenseNet169	90.40%	Para la clasificación de COVID-19 de tomografías computarizadas.	(Martinez, 2020).
16	DenseNet201	87.7%	Para la clasificación de la caña de azúcar utilizando una red neuronal convolucional.	(Virnodkar, Patil, & Jha, 2020)
17	GoogleNet	98%	Para el reconocimiento de plagas de cultivos con precisión.	(Yanfen, Hanxiang, Minh, & Abolghasem, 2020)
18	NASNetLarge	93.27%	evaluar la posibilidad de desarrollar un sistema de detección de pólipos colorrectales.	(Jinsakul, Tsai, Tsai, & Wu, 2019)
19	EfficientNetB0	77%	Para clasificación e identificación de sarcomas de tejidos blandos	(Lo, y otros, 2020)

20	EfficientNetB1	86.2%	Enfoque de fusión de tres niveles utilizando múltiples redes profundas para lesiones cutáneas.	(Amirreza, y otros, 2020)
21	EfficientNetB2	79.8%	Estudio comparativo de la EfficientNet con otras CNN existentes en ImageNet.	(Mingxing & Quoc, 2020)
22	EfficientNetB3	80.50%	Estudio comparativo de la EfficientNet con otras CNN existentes en ImageNet.	(Mingxing & Quoc, 2020)
23	EfficientNetB4	82.50%	Estudio comparativo de la EfficientNet con otras CNN existentes en ImageNet.	(Mingxing & Quoc, 2020)
24	EfficientNetB5	83.80%	Estudio comparativo de la EfficientNet con otras CNN existentes en ImageNet.	(Mingxing & Quoc, 2020)
25	EfficientNetB6	84.0%	Estudio comparativo de la EfficientNet con otras CNN existentes en ImageNet.	(Mingxing & Quoc, 2020)

26	VGGNet	91.83%	La CNN profunda se aplicó en la identificación de enfermedades de las plantas	(Chen, y otros, 2020)

Fuente: Elaboración Propia.

Con base en el ranking de la tabla número de comparación de métodos de clasificación anteriormente mostrada se seleccionó a 6 redes convolucionales por criterios con los que obtuvieron un mejor desempeño en términos de precisión y por el tipo de imágenes que procesaron dando prioridad a las que procesaron las palmas de las manos como la VGG16, MobileNetV2 y otras que obtuvieron acuracy alto como VGG19, Xception, ResNet50 y DenseNet121 en sus investigaciones ya citadas en la tabla anterior.

Tabla 12

Selección de 6 redes neuronales con mejor desempeño.

N	Modelo	Precisión	Criterio de selección
1	VGG16	100%	Mejor precisión Tipo de imagen
2	MobileNetV2	100%	Mejor precisión Tipo de imagen
3	VGG19	99.90%	Mejor precisión
4	Xception	99.50%	Mejor precisión
5	ResNet50	99.0%	Mejor precisión
6	DenseNet121	98.0%	Mejor precisión

Fuente: Elaboración Propia

3.3.3 Diseñar un nuevo método en base a las técnicas seleccionadas.

Para desarrollar el método de identificación de personas se procedió a hacer el entrenamiento de las 6 redes neuronales seleccionadas VGG16, MobileNetV2, VGG19, Xception, ResNet50 y DenseNet121 las cuales dieron como resultados los modelos de cada red neuronal. Estas mismas fueron evaluadas para conocer la red neuronal de mejor precisión de detección de huellas palmarias y con esto se eligió nuestro nuevo método de identificación.

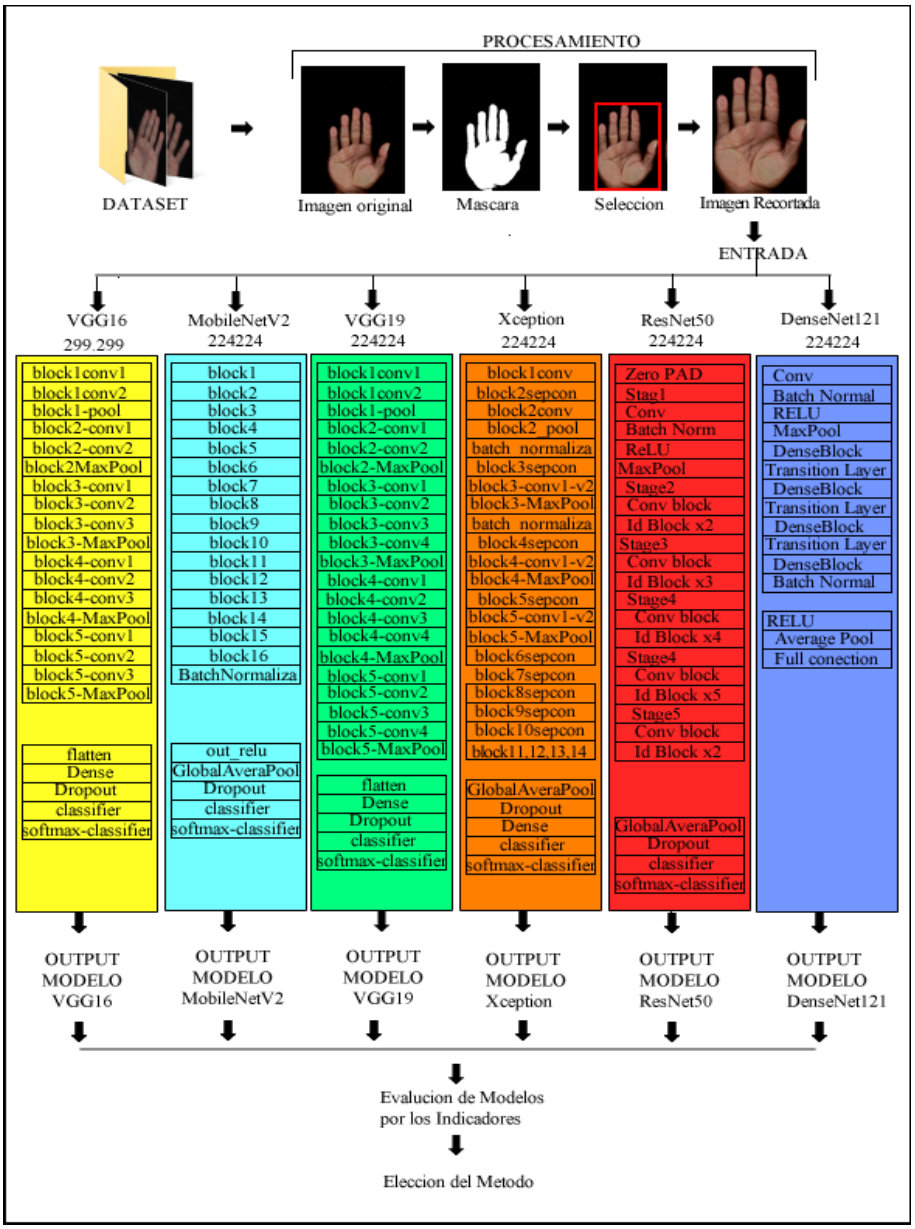

Figura 29. Pasos para elección del nuevo método de identificación

Fuente: Elaboración propia

3.3.4 Codificación en lenguaje de programación del método

Al iniciar a construir el método se procedió a preparar las imágenes del dataset de manera que puedan ser procesadas por las redes convolucionales es por ello que se creó el archivo Python align_dataset.py que se muestra y explica a continuación:

```
def crop_scan(img):  # funcion que aplica a la imagen el algoritmo de segmentacion
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  # se crea una copia en esca
la de grises
    blur = cv2.GaussianBlur(gray, (5, 5), 1)  # se le aplica un filtro gaussiano  ruido
    thresh = cv2.threshold(blur, 0, 255, cv2.THRESH_OTSU + cv2.THRESH_BINA
RY)[1]  # se hace un threshold las siguientes 2 operaciones morfologicas son para el
iminar ruido y artefactos en la mascara
    clean = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel=cv2.getStructurin
gElement(cv2.MORPH_ELLIPSE, (5, 5)))
    clean = cv2.morphologyEx(clean, cv2.MORPH_CLOSE, kernel=cv2.getStructuri
ngElement(cv2.MORPH_ELLIPSE, (5, 5)))
    cnts = cv2.findContours(clean, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_
SIMPLE)[0]  # serching Bordes
    cnt = max(cnts, key=lambda c: cv2.contourArea(c))  # se elige el contorno de may
or tamaño
    x, y, w, h = cv2.boundingRect(cnt)  # se obtienen las coordenadas del rectangulo
que contiene la region de interes
    return img[y:y + h, x:x + w]  # se devuelve la imagen original recortada
```

Figura 30. Segmentación de imágenes, threshold, búsqueda de contornos.

Fuente: Elaboración propia

Creamos los directorios la carpeta de testeo y listamos todos los archivos

```
def align_split_dataset(target_dir, new_dir, subdir, samples_to_test):
    make_dir(new_dir)  # creamos la nueva carpeta
    train_dir = os.path.join(new_dir, "train")
    make_dir(train_dir)  # creamos la carpeta de entrenamiento
    test_dir = os.path.join(new_dir, "test")
    make_dir(test_dir)  # creamos la carpeta de testeo
```

Figura 31 Creación de los directorios

Creamos las nuevas carpetas donde y asignamos la ruta donde van las huellas palmarias segmentadas.

```
        class_name = os.path.basename(c)
    make_dir(os.path.join(train_dir, class_name))  # Ruta donde van las segmentadas
            new_class_dir = os.path.join(test_dir, class_name) if n < samples_to_test els
e os.path.join(train_dir, class_name)
img = cv2.imread(img_path)  # carga de imagen original del disco
            total += 1
     if img is None:  #si hay algun problema con la imagen, la ignoramos y seguimos
                print("ERROR LOADING IMAGE:", img_path)
                continue
     )
```

Figura 32 definición de rutas de nuevas imágenes segmentadas

Luego de que los directorios rutas y imágenes estuvieron listas se creó el archivo Train.py el cual carga el dataset al iniciar el proceso, se filtró las alertas de que muestra TensorFlow y se importaron los modelos.

Se creó los directorios que necesitamos para guardar los modelos, gráficas, resultados y se procesó las imágenes dándole transformaciones para aumentar la robustez del modelo.

```
def make_dirs():
    dirs = ["./models", "./graphs"]
    for d in dirs:
        if not os.path.isdir(d):  # si no existe el directorio, se crea
            os.mkdir(d)
def train(
        name,  # id
        train_data_dir, test_data_dir, augmentations,  # datos
        batch_size, height, width, architecture, dropout,  # modelo
        optimizer, max_epochs, early_stop_patience  # entrenamiento
):
    seed = 911  # seteamos la semilla para hacer el código reproducible
    make_dirs()  # creamos los directorios
    target_size = (height, width)  # C

    # PREPROCESAMIENTO
    train_datagen = ImageDataGenerator(
        **augmentations,  # transformaciones para aumentar la robustez del modelo
        rescale=1.0 / 255.0  # reescalamos la intensidad de los pixeles entre [0-1])
    test_datagen = ImageDataGenerator(
        rescale=1.0 / 255.0,  # reescalamos la intensidad de los pixeles entre [0-1] )
```

Figura 33 Ruta donde van las imágenes segmentadas nuevas

Se implementó generadores de datos desde el disco con las clases inferidas a partir de los nombres de las carpetas.

```
    print("Training:", end=" ")
    train_generator = train_datagen.flow_from_directory(
        train_data_dir,  #carpeta donde estan las imágenes entrenamiento
        classes=None,  # dejamos que infiera el nombre de las clases
        seed=seed,  # le damos la semilla para randomizaciones
        target_size=target_size,  # damos tamaño para resizear la imagen
        batch_size=batch_size,  # le damos el tamaño de lote de procesamiento
        class_mode='categorical',  # le decimos que use one-
hot encoding para las etiquetas (multi-clase)
        color_mode='rgb',  #decimos que cargue en RGB las imágenes,3 canales
        interpolation='bicubic'  # interpolación empleada en el resize
    )
n_classes = train_generator.num_classes  # guardamos en una variable la cantidad de
clases que tiene el dataset
    print("Validation:", end=" ")
    test_generator = test_datagen.flow_from_directory(
        test_data_dir,  # carpeta donde se encuentran las imagenes de testeo
        classes=None,  # dejamos que infiera el nombre de las clases .
        seed=seed,  # le damos la semilla para randomizaciones
        target_size=target_size,  #damos el tamaño al que  resizeara la imagen
        batch_size=batch_size,  # le damos el tamaño de lote de procesamiento
        class_mode='categorical',  # le decimos que use one-
hot encoding para las etiquetas (multi-clase)
        color_mode='rgb',  # le decimos que cargue en RGB las imagenes, o sea en 3 can
        interpolation='bicubic'  # interpolación empleada en el resize
```

Figura 34.G*eneradores de datos*

Se construyó la red neuronal sin las capas de clasificación para agregarle las nuestras.

```
# construir la red neuronal
  base_model = architecture(
      include_top=False,  # le sacamos la "cabeza" de clasificacion
      weights="imagenet",  # partimos de los pesos del modelo
      pooling=None,  # el pooling hacemos abajo a mano para que quede claro
      input_shape=(height, width, 3), #dimensiones de nuestro tensor de imagen
      backend=K,  # temas internos
      layers=L,  # temas internos
      models=M,  # temas internos
      utils=U  # temas internos  )
  out = base_model.output #tomamos referencia a la capa final de modelo base
  if architecture in (VGG16, VGG19):# s es VGG utilizamos otra cabeza
      out = Flatten(name='flatten')(out)  # operacion que conserva todos los pixeles y
los "aplana" a 1D
      out = Dense(1024, activation='relu', name='fc1')(out)  # capa de 1024 neuronas
(este valor puede tunnearse)
  else:
      # para todas las demás utilizamos global average pooling
      out = GlobalAveragePooling2D(name='g.a.p.')(out)  # hacemos un average-
pooling para convertir la data a 1D
  if dropout:  # agregamos dropout para reducir overfitting
      out = Dropout(dropout, name='dropout_{:.2f}'.format(dropout))(out)
  out = Dense(n_classes, name='classifier')(out)  # agregamos la capa de clasificaci
on con nuestro num de clases
  out = Activation('softmax', name='softmax_classifier')(out)  # salida softmax para
convertir a probabilidad
  model = Model(inputs=base_model.input, outputs=out)  #se cierra el modelo
```

Figura 35. Construcción de la red neuronal introducción de parámetros

Se procedió a imprimir el sumario de la arquitectura y compilar el modelo con el optimizador pasándole las métricas.

```
  # impricesion
   model.summary()
   # compilar modelo
   model.compile(optimizer=optimizer, loss='categorical_crossentropy', metr
ics=['categorical_accuracy'])
   # callback para reducir el learning rate a la mitad cuando no progresa
   learning_rate_reduction = ReduceLROnPlateau(monitor='val_loss', mode
='min',  # queremos minimizar el loss de testeo
  patience=3,  # le damos una paciencia de 3 epocas antes de reducirlo
  verbose=1,  # que avise cuando lo hace
  factor=0.5,  # cada reduccion es a la mitad
  min_lr=1e-7,  # minimo learning rate)
   # callback para detener el entrenamiento automaticamente cuando haya dej
ado de progresar por varias epocas
   earlystop = EarlyStopping(
       monitor='val_loss', mode='min',  # queremos minimizar el loss de testeo
       patience=early_stop_patience,  # le seteamos la paciencia antes de finali
zar el entrenamiento
       restore_best_weights=True,  # nos quedamos con el modelo que tuvo m
enos loss de validacion
       verbose=1,  # que avise cuando termine
   )
   # definimos los callbacks en un vector para pasarle al entrenamiento
   callbacks = [learning_rate_reduction, earlystop]
```

Figura 36 imprimir el sumario de la arquitectura y compilar el modelo

Se entrenó el modelo definiendo el generador para entrenamiento.

```
# entrenar modelo
    history = model.fit_generator(
        train_generator,  # generador que definimos para entrenamiento
        epochs=max_epochs,  # cantidad maxima de epocas (esperamos que c
orte antes igualmente)
       validation_data=test_generator,  # generador que definimos para testeo
        callbacks=callbacks,  # callbacks que definimos
        steps_per_epoch=train_generator.samples // batch_size,  # una epoca
corresponde a una pasada completa
        validation_steps=test_generator.samples // batch_size,  # una epoca co
rresponde a una pasada completa
        verbose=1  # imprimir progreso
    )
```

Figura 37 Entrenamiento del modelo

Se evaluó algunos modelos por comportamiento particular de BatchNormalization en keras.

```
 # la evaluacion final
    K.set_learning_phase(0)  # salir de la fase de entrenamiento
    # definimos la cantidad de pasos de forma tal que se evaluen todos los eje
mplos
    s = test_generator.samples
    steps = (s / batch_size) if s % batch_size == 0 else (s // batch_size + 1)


    print("\nEvaluation\n")
    loss, acc = model.evaluate_generator(test_generator, steps=steps, verbose
=1)  # evaluamos el modelo final
    print("\nloss: {:.4f} | acc: {:.1f}%".format(loss, acc * 100))
```

Figura 38. Evaluación final.

Se graficó el historial de entrenamiento con un nombre para identificarlo con la exactitud obtenida, guardando la estructura del modelo y guardando los pesos aprendidos.

```
# graficar el historial
    plot_history(history.history, path="./graphs/{}_acc{:.2f}.png".format(nam
e, acc))
    plt.clf()
    # guardar estructura del modelo en formato json
    model_json = model.to_json()
    with open("./models/{}_architecture.json".format(name), "w") as json_file
:
        json_file.write(model_json)
    # guardar los pesos aprendidos en formato hdf5
    model.save_weights("./models/{}_weights.h5".format(name))
```

. Figura 39 graficó el historial de entrenamiento en reportes .png

Se guardó la lista de etiquetas y se convirtió la salida de la red neuronal a los nombres de las clases.

```
 # guardar  de etiquetas
    labels = list(train_generator.class_indices)
    with open("./models/{}_labels.json".format(name), "w") as json_file:
        json.dump(labels, json_file)
if __name__ == "__main__":
    # transformaciones en entrenamiento, modificar valores y/o comentar las q
ue no se quieran utilizar
    aug = {
        "brightness_range": [0.8, 1.2],  # variaciones aleatorias en brillo
        "rotation_range": 30,  # rotaciones aleatorias en ambas direcciones
        # "width_shift_range": 0.1,  # desplazamiento horizontal aleatorio (con
nuestro alineamiento no seria necesario)
        "height_shift_range": 0.1  # desplazamiento vertical aleatorio (puede ser
vir para hacerlo robusto al brazo }
```

Figura 40. Guardar la lista de etiquetas.

Se asignó el optimizador y configuró el learning rate inicial, dando dimensiones ideales a las siguientes redes neuronales VGG16, MobileNetV2, VGG19, ResNet50, DenseNet121 : (224, 224, 3) Xception: (260, 260, 3) para cada vez que se entrene cada red neuronal.

```
# optimizador
    opt = Adam(lr=5e-4)
    train(
        name="manos_alineadas_derecha_MobileNetV2",  # nombre
        # datos
        train_data_dir="C://Users//BARNY//Desktop//barny1//dataset_align_
right//train",  # ruta al split de entrenamiento
        test_data_dir="C://Users//BARNY//Desktop//barny1//dataset_align_ri
ght//test",  # ruta al split de testeo
        augmentations=aug,  # diccionario con transformaciones a las imagen
es durante el entrenamiento
        # modelo
        batch_size=4,  # tamaño de lote para entrenar
        height=224,  # cantidad de filas en la matriz imagen
        width=224,  # cantidad de columnas en la matriz imagen
        architecture=MobileNetV2,  # vgg16, vgg19, resnet50, efficientnetb0,
 efficientnetb2, densenet121      xception resnet152v2  mobilenetv2
        dropout=0.5,  # fracción de dropout
        optimizer=opt,  # optimizador que ajusta los gradiente
        max_epochs=200,  # numero de épocas máximo permitido
        early_stop_patience=10  # paciencia en epocas para finalizar el entren
amiento cuando el modelo deje de mejorar
    )
    K.clear_session()  # limpiamos la sesion de keras antes de salir
```

Figura 41 configuración de learning rate, dando dimensiones para redes neuronales

3.3.5 Evaluar el resultado del nuevo método planteado.

En esta Investigación para conocer los resultados se creó un script Python para evaluar los modelos que fueron creados por los entrenamientos de las redes convolucionales sobre las imágenes de test.

```
import numpy as np
import json
from time import time
import cv2
def crop_scan(img):  #funcion que aplica a la imagen el algoritmo segmenta
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  # Escala grises
    blur = cv2.GaussianBlur(gray, (5, 5), 1)  #  filtro gaussiano
    thresh = cv2.threshold(blur, 0, 255, cv2.THRESH_OTSU + cv2.THRESH_BINA
RY)[1]  # se genera una mascara binaria
    clean = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel=cv2.getStructuri
ngElement(cv2.MORPH_ELLIPSE, (5, 5)))
    clean = cv2.morphologyEx(clean, cv2.MORPH_CLOSE, kernel=cv2.getStructuri
ngElement(cv2.MORPH_ELLIPSE, (5, 5)))
    cnts = cv2.findContours(clean, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX
_SIMPLE)[0]  # se buscan contornos en la mascara
    cnt = max(cnts, key=lambda c: cv2.contourArea(c))  # el contorno
 x, y, w, h = cv2.boundingRect(cnt)  #se obtienen las coordenadas del rectangulo
    return img[y:y + h, x:x + w]  #se devuelve la imagen  recortada
def build_model(path_to_model_arch, path_to_model_weights, path_to_model_lab
els, target_size):
    st = time()
```

Figura 42 Evaluación del resultado del nuevo método planteado.

Se construyo el modelo, cargar pesos y etiquetas.

```
print("LOADING MODEL...")
with open(path_to_model_arch, "r") as model_json:  # abre el json con la arquitectura del modelo y lo construye
    model = model_from_json(model_json.read())
model.load_weights(path_to_model_weights)  # carga los pesos guardados en el entrenamiento
with open(path_to_model_labels, "r") as json_file:  # abre la lista de etiquetas para convertir output del modelo
    labels = json.load(json_file)  # en los nombre que tienen las etiquetas del data
# predecir con data random para 'precalentar' el modelo (siempre la primer prediccion es lenta)
model.predict(np.random.rand(1, *target_size, 3)).astype(np.float32)
et = time() - st  # tomar el tiempo de construccion modelo
```

Figura 43 Construcción y carga de pesos y etiquetas

Para la reprocesamiento de la imagen tiene que ser reprocesada de la misma manera que se hizo para entrenar

```
image = cv2.imread(path_to_image)
image = crop_scan(image)  # usamos el mismo algoritmo
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)  # convertimos la imagen a rgb como se manejaron en el train
image = cv2.resize(image,
                   target_size[::-1],  # resizeamos la imagen al tamaño que el modelo tiene como input
                   interpolation=cv2.INTER_CUBIC)  # usamos interpolacion bicubica igual que en el entrenamiento
image = image.astype(np.float32)  # convertimos el tipo de dato a float32 para que pueda contener valores decimales
image /= 255.0  # reescalamos los pixeles en el rango [0, 1] igual que en el entrenamiento
```

Figura 44 Reprocesamiento de valores de la misma manera de entrenar

Se Cargó el modelo y las etiquetas

```
    # le damos las tres rutas que corresponden al modelo que queremos utilizar (a
rquitectura, pesos y etiquetas)
    path_to_model_arch="./models/manos_alineadas_derecha_EfficientNetB2_ar
chitecture.json",
    path_to_model_weights="./models/manos_alineadas_derecha_EfficientNetB
2_weights.h5",
    path_to_model_labels="./models/manos_alineadas_derecha_EfficientNetB2_
labels.json",
    path_to_image="./pruebas/persona2_img039.jpg",  # imagen sin preprocesar
(que no haya estado en el split train)
    target_size=(260, 260)  # las mismas dimensiones de imagen con las que se e
ntreno el modelo
```

Figura 45 carga de modelos pesos y etiquetas previamente creados en el entrenamiento

Al ejecutar el este archivo dio como resultados lo mostrado a continuación en la siguiente matriz de confusión.

Tabla 13

Resultados de Matriz de confusión

	FP	FN	TP	TN
VGG16	0.25	0.25	0.75	98.75
MobileNetV2	0.05	0.05	0.95	98.95
VGG19	0.27	0.27	0.73	98.73
Xception	0.03	0.03	0.97	98.97
ResNet50	0.01	0.01	0.01	98.99
DenseNet121	0.02	0.02	0.98	98.98

El script también imprime en pantalla las manos o clases donde erraron en la predicción las redes convolucionales.

Errores de predicción del modelo VGG16:

```
1) sample of class '01' predicted as '69'
2) sample of class '02' predicted as '56'
3) sample of class '03' predicted as '20'
4) sample of class '06' predicted as '10'
5) sample of class '07' predicted as '39'
6) sample of class '09' predicted as '19'
7) sample of class '10' predicted as '20'
8) sample of class '11' predicted as '18'
9) sample of class '16' predicted as '17'
10) sample of class '17' predicted as '68'
11) sample of class '20' predicted as '80'
12) sample of class '21' predicted as '81'
13) sample of class '22' predicted as '26'
14) sample of class '27' predicted as '20'
15) sample of class '29' predicted as '89'
16) sample of class '30' predicted as '90'
17) sample of class '42' predicted as '39'
18) sample of class '44' predicted as '96'
19) sample of class '50' predicted as '49'
20) sample of class '53' predicted as '56'
21) sample of class '55' predicted as '61'
22) sample of class '58' predicted as '49'
23) sample of class '60' predicted as '61'
24) sample of class '71' predicted as '11'
25) sample of class '82' predicted as '62'
```

Figura 46 Salida de consola muestra errores de la red neuronal VGG16

Errores de predicción del modelo MOBILENETv2

```
1) sample of class '02' predicted as '73'
2) sample of class '06' predicted as '10'
3) sample of class '20' predicted as '80'
4) sample of class '30' predicted as '90'
5) sample of class '53' predicted as '01'
```

Figura 47 salida de consola muestra los errores de la red MobilenetV2

Errores de predicción del modelo VGG19

```
1) sample of class '01' predicted as '69'
2) sample of class '02' predicted as '30'
3) sample of class '03' predicted as '55'
4) sample of class '06' predicted as '10'
5) sample of class '07' predicted as '57'
6) sample of class '09' predicted as '19'
7) sample of class '10' predicted as '60'
8) sample of class '11' predicted as '18'
9) sample of class '12' predicted as '25'
10) sample of class '16' predicted as '51'
11) sample of class '17' predicted as '50'
12) sample of class '18' predicted as '55'
13) sample of class '20' predicted as '80'
14) sample of class '21' predicted as '81'
15) sample of class '22' predicted as '15'
16) sample of class '27' predicted as '20'
17) sample of class '29' predicted as '89'
18) sample of class '30' predicted as '90'
19) sample of class '44' predicted as '43'
20) sample of class '50' predicted as '11'
21) sample of class '52' predicted as '95'
22) sample of class '53' predicted as '72'
23) sample of class '55' predicted as '61'
24) sample of class '56' predicted as '58'
25) sample of class '71' predicted as '11'
26) sample of class '84' predicted as '44'
27) sample of class '97' predicted as '58'
```

Figura 48 Consola muestra los errores de la red neuronal VGG16

Errores de predicción del modelo Xception

```
model mistakes:
1) sample of class '01' predicted as '69'
2) sample of class '02' predicted as '09'
3) sample of class '30' predicted as '90'
```

Figura 49 salida de consola muestra errores de red neuronal Xception

Errores de predicción del modelo ResNet50

```
1) sample of class '30' predicted as '90'
```

Figura 50 Salida de consola muestra errores de la red neuronal ResNet50

Errores de predicción del modelo DenseNet121

```
1) sample of class '05' predicted as '85'
2) sample of class '30' predicted as '90'
```

Figura 51 salida de consola muestra errores de red neuronal DenseNet121

IV. CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

En base a los resultados, se llegó a las siguientes conclusiones:

Para el desarrollo del método es importe la etapa de preparación del Dataset, en esta etapa, la adquisición de las imágenes se realizó con un dispositivo de tipo scanner alta resolución donde se puede apreciar la mayor cantidad de detalles como las líneas palmarias que permite converger a las redes neuronales.

Para seleccionar los métodos de clasificación de imágenes se realizó por medio de una revisión de la literatura de las redes neuronales existentes hasta la fecha encontrando 26 redes neuronales, eligiendo las 6 con mejores resultados para este propósito por medio de un ranking de resultados de precisión en la literatura.

La codificación del método fue realizada en lenguaje Python 3.7.9 puesto que en estudios previos esto se realizado en este lenguaje. En esta investigación se realizó en dos etapas, en la primera etapa se realizó la programación para alinear las imágenes digitales del dataset y en la segunda etapa se realizó la codificación para el ingreso del dataset al entrenamiento de las redes neuronales que permiten la creación del modelo de cada una de las 6 arquitecturas usadas.

Como parte del desarrollo del nuevo método se realizó la codificación de un script para medir los resultados de los 6 modelos arrojados por las redes neuronales las cuales permitieron concluir el desarrollo del método, se tomó a la red neuronal Resnet50 que es la más adecuada para la tarea de identificar personas por el mejor resultado con un 99% de acuraccy quedando esta como un paso más del método propuesto.

Cabe resaltar el trabajo previo de Mejia, Flores, Tuesta y Forero en el papper “New method for subject identification based on palm print” obtuvieron 100 % de exactitud con imágenes de 50 personas. Y en este trabajo 99 % de exactitud esto se debe a la muestra fue mayor de 100 personas y 3 imágenes de cada persona. Quedando mejor representada la muestra en esta investigación.

RECOMENDACIONES

Se recomienda que para adquirir imágenes se realice con scanner que cuente con un mínimo de profundidad 24 de bit, y se edifique una estructura que permita control de la luz en el ambiente y eso permita que el proceso de segmentación de la imagen sea el mejor.

En la preparación del dataset se podría adquirir las manos sin que salga el brazo de la persona, o entrenar un modelo de detección de landmarks para alinear la mano.

Se recomienda trabajar con una cantidad mayor de imágenes digitales por cada mano, y lo ideal es estandarizar la que se utilizará para la identificación, por ejemplo, sólo considerar la derecha y tomar 10 imágenes digitales de huellas palmarias de los cuales 7 se emplean en 'train' y 3 en 'test', en lugar de 2 y 1).
En la codificación del método encontrar los hiperparámetros adecuados, en general conviene elegir 1 red neuronal que se vea que da buen resultado y a partir de ella configurar los hiperparámetros, la data augmentation y el preprocesamiento de las imágenes hasta lograr los mejores resultados.

Se recomienda trabajar la implementación de los algoritmos con Python Python 3.7.9 dado que facilita la implementación de las librerías de keras versión 2.0 de esta manera se pueda soportar en la librería de TensorFlow 2.0.

REFERENCIAS

Aashni, H., Archanasri, S., Nivedhitha, A., Shristi, P., & Jyothi, S. (2017). Hand Gesture Recognition for Human Computer Interaction. *Procedia Computer Sciencie, 115*, 367-374. Obtenido de https://www.sciencedirect.com/science/article/pii/S1877050917319130

Amirreza, M., Gerald, S., Chunliang, W., Georg, D., Rupert, E., & Isabella, E. (2020). Transfer learning using a set of multiple scales and networks for the classification of skin lesions. *Science Direct, 193*. doi:105475

Aurelia, M., Vincent, C., & Díaz, S. (2019). MobileNet convolutional neural networks and support vector machines for palm print recognition. *Science Direct, 157*, 110-117. Recuperado el 15 de Octubre de 2020, de https://www.sciencedirect.com/science/article/pii/S1877050919310658

Beltrán, J., & Arevalo, W. (2019). *SELECCIÓN DE MODELO DE RED NEURONAL PROFUNDA (DEEP LEARNING) ii ÓPTIMO PARA EL RECONOCIMIENTO DE ENFERMEDADES USANDO IMÁGENES DE RAYOS X*. Obtenido de http://repositorio.ucundinamarca.edu.co/bitstream/handle/20.500.12558/3006/LIBRO.pdf?sequence=1&isAllowed=y

Bertino, C., & Orozco, S. (2018). *Diseño e implementación de un sistema Biométrico de reconocimiento palmar*. Obtenido de Library: https://1library.co/document/y6e4oxnz-diseno-implementacion-sistema-biometrico-reconocimiento-palmar.html

Blanco, C. (2018). *Google.* Obtenido de http://oa.upm.es/50271/1/TFG_CRISTINA_BLANCO_GARRIDO.pdf

Chen, J., Chen, J., Zhang, D, Sun, Y., & Nanehkaran. (2020). Using deep transfer learning for image-based plant disease identification. *Science Direct, 173*. doi:105393

Dalia, E., Aboul, E., & Hassan, A. (2020). A Deep Learning Architecture Optimized for COVID-19 Disease Diagnosis Based on Gravitational Search Optimization. (S. Direct, Ed.) *Apllied SoftComputing*, 52-70. doi:106742

Felipe, E., & Suárez, S. (2015). *Implementación de algoritmos de procesamiento de imágenes en FPGA.* Informe de tesis, Instituto Politécnico Nacional-Centro de investigación en Computación, México, D.F. Recuperado el 23 de octubre de 2020, de http://148.204.63.111/SABERv3/Repositorios/webVerArchivo/26041

Gamarra, C, & Ríos, M. (2017). *Facultad de Ingeniería Electrónica*. Obtenido de https://repository.usta.edu.co/bitstream/handle/11634/10680/2018Gamarracamilo.pdf?sequence=1

Gampala, S., Gurugubelli, S., & Hari, B. (2018). Artificial neural network and regression modeling to study the effect of reinforcement and deformation on volumetric wear of aluminum matrix composites. *Boletín de la Sociedad Española de Cerámica y vidrio, 57*(3), 91-100. doi:10.1016/j.bsecv.2017.09.006

Gaurav, J., Amit, K., & Ravinder, N. (2018). Multifunctional Fusion for Unrestricted Palm Print Authentication, Biomedical Instrumentation and Signal Processing Laboratory,. (S. Direct, Ed.) *Computers & Electrical Engineering, 72*, 53-78. doi:10.1016/j.compeleceng.2018

Gopal, G., Srivasta, S., Bhardwaj, S., & Bhargava, S. (2016). Fusion of impression of palmeral phalanges with palmar impression and dorsal vein of the hand. (ELSEVIER, Ed.) *Procedia Computer science, 47*, 12-20. doi:10.1016/j.asoc.2016

Guarav, J., & Amit, K. R. (2018). Multifuncional Fusion for unrestricted palm print authentication. (SciencieDirect, Ed.) *computers and Elecrical Engineering*, 53-78. doi:10.1016/j.compelence.2018009.006

Hafiz, T., Lkram, U., Saliha, Z., Syed, Z., Abd, R., & Chan, B. (2019). Palmprint Recognition using Robust Template Matching, Birla Institute of Technology and Science. *Computers and Electronics in Agriculture, 167*, 75-86. doi:10.1016/j.compag.2019

Hernández, S. (2010). *Metologìa de la Investigaciòn.* (INTERAMERICANA, Ed.) Mèxico: McGrawGill.

Hussein, Ibrahim, & Taha. (2020). Framework for deep pneumonia using deep Learnig models based on chest X-Ray imaging. *ResearGate*. Recuperado el 16 de octubre de 2020, de

https://www.researchgate.net/publication/344058411_Deep-Pneumonia_Framework_Using_Deep_Learning_Models_Based_on_Chest_X-Ray_Images

Jinghe, W., Younis, I., Siyu, Q., Haibin, W., Guozhu, L., Qingkui, Y., . . . Junwei, S. (2020). Analyzing the impact of soft errors in VGG networks implemented on GPUs. *Microelectronics Reliability, 110*, 6-20. doi:10.1016/j.microrel.2020.113648

Jinsakul, N., Tsai, C., Tsai, C., & Wu, P. (2019). *Mathworks.* Obtenido de https://www.sciencedirect.com/science/article/abs/pii/S0168169919313638

Ketan, J., Vikas, T., Chitransh, B., & Chaitanya, B. (2019). Clasificación robusta de imágenes deportivas mediante InceptionV3 y redes neuronales. *Science Direct, 154*, 369-376. Recuperado el 18 de Octubre de 2020, de https://www.sciencedirect.com/science/article/pii/S1877050919308221

Lo, S., Nitsh, J., Bauer, S., Beltram, S., Goetz, M., Hamacher, R., . . . Schacherer, D. (2020). IA 1659P based classification approach identifiles FNCLCC grade 3 soft tissue sarcomas. *Science Direct, 169*. doi:10.1016

Maeda, V., Guerrero, C., Olvera, C., Araiza, M., Espinoza, G., & Bordón. (2018). Convolutional neural networks for detection and classification of plant diseases based on digital imagenes. *Biológico Agropecuaria Tuxpan*, 275-281. Recuperado el 18 de agosto de 2020, de http://ricaxcan.uaz.edu.mx/jspui/bitstream/20.500.11845/1910/1/2018%20Arbitrado%20Maeda.pdf

Martinez, A. (2020). Classification of COVID-19 in CT scans usin mutisource transfer learning. *ReseaeGate*. doi:10.13140

McAllister, P., Zheng, H., Bond, R., & Moorhead, A. (2018). Combining deep residual network functions with supervised machine learnig algorithms to classify diverse food image data sets. *ReseasrchGate*. Recuperado el 12 de octubre de 2020, de https://www.researchgate.net/publication/323249372_Combining_deep_residual_network_features_with_supervised_machine_learning_algorithms_to_classify_diverse_food_image_datasets

Meenaxi, M., Meena, S., Chairtra, K., & Shantala, G. P. (2020). Okra-ladies finger grading and grading using deep learning. *Science Direct, 171*, 2380-2389.

Recuperado el 12 de Octubre de 2020, de https://www.sciencedirect.com/science/article/pii/S1877050920312503

Mehmet, O., Nurullah, C., & Tulay, Y. (2019). Painter Classification on the Novel Art Painting Data Set Via Latest Deep Neural Networks. *Sciente Direct, 154*, 369-376. Recuperado el 9 de Octubre de 2020, de https://www.sciencedirect.com/science/article/pii/S1877050919308221

Mejía, I., Antón, R., Flores, N., Tuesta, V., & Forero, M. (2020). New method for subject identification. *browse Proceedings, 11510*, 5-15. doi:10.1117/12

Mingxing, T., & Quoc, L. (2020). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. (arXiv, Editor) Obtenido de https://arxiv.org/pdf/1905.11946.pdf

Mohammad, & Abolfazl. (2020). A modified deep convolutional neural network for detecting COVID-19 and pneumonia from chest X-ray images based on the concatenation of Xception and ResNet50V2. *Science Direct, 19*. doi:10.100360

Nishant. (2018). Machine learning with TensorFlow. *Sematicsholar*. Recuperado el 14 de octubre de 2020, de https://www.semanticscholar.org/paper/Machine-Learning-with-TensorFlow-Shukla/4fc2c857a093aa833f8624bc7f21b4b2c78ef9c3#paper-header

Ojeda, J. (2015). *Elaboración de un laboratorio virtual para mejorar el aprendizaje de filtros digitales en DSP en la Facultad Técnica para el Desarrollo de la Universidad Católica de Santiago de Guayaquil.* Guayaquil: Repositorio Diogital UCSG. Recuperado el 1 de Noviembre de 2020, de http://repositorio.ucsg.edu.ec/handle/3317/1723

Poonam, P., Pawan, A., & Vijayendra, S. (2020). Palmprint Recognition using Robust Template Matching. *Procedia Computer Science, 167*, 727-736. Recuperado el 05 de setiembre de 2020, de https://www.sciencedirect.com/science/article/pii/S1877050920308048

Rodes, F. (2018). Cuaderno de prácticas de identificación forense. En F. Rodes, *Cuaderno de prácticas de identificación forense* (págs. 15-32). España: Cegal. Obtenido de https://publicaciones.ua.es/files/detalles/54456269788497174534_Fragment.pdf

Rubio, J., Hernández, J., Ávila, F., Stein, J., & Meléndez, A. (2016). Sensor System Based in Neural Networks for the Environmental Monitoring. (S. Direct, Ed.) *Ingeniria, Investigación y Tecnología*, 211-222. doi:10.1016

Sancho, C. (2018). Breve Historia de la inteligencia artificial. *Occidente, 446*, 19-33. Recuperado el 02 de agosto de 2020, de https://dialnet.unirioja.es/servlet/articulo?codigo=6503791

Shashi, B., & Nidhi, N. (2016). Comparative analysis of the palm print recognition system with thw repeated line tracking method. *Procedia Computer science, 92*, 578-582. Recuperado el 15 de Agosto de 2020, de https://www.sciencedirect.com/science/article/pii/S1877050916316416

Thiago, A., Pinto, A., Ventura, P., Neves, C., Biron, D., Junior, A., . . . Rodrigues, P. (2020). Automatic detection and classification of cells in honeycomb using deep learning. *Science Direct, 170*. Recuperado el 17 de Octubre de 2020, de https://www.sciencedirect.com/science/article/abs/pii/S0168169919307690

Tomoyuki, F., Yuka, Y., Jun, O., & Mori, M. K. (2020). Deep Learning Approach with Convolutional Neural Network for Classification of Dynamic Contrast Breast Magnetic Resonance Maximum Intensity Projections. *Science Direct, 75*, 1-8. Recuperado el 15 de octubre de 2020, de Deep Learning Approach with Convolutional Neural Network for Classification of Dynamic Contrast Breast Magnetic Resonance Maximum Intensity Projections

Tuesta, V., Alcarazo, F., Mejía, I., & Forero, M. (2020). Automatic classification of citrus. *Browse Proceedings, 11510*, 2-10. doi:10.1117/12.2566888

Vacacela, U., Fiama, C., & Mantilla, A. (2015). *Procesamiento Digital de Imágenes Aplicado al Control de Calidad para la Estación de Almacenamiento Festo de la EIECRI.* Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo. Riobamba, Ecuador: Escuela Superior Politécnica de Chimborazo. Recuperado el 25 de octubre de 2020, de http://dspace.espoch.edu.ec/handle/123456789/4453

Viera, G. (2018). *Repositorio Institucional PIRHUA.* Universidad de Piura. Facultad de Ingeniería. Piura, Perú. Piura: Universidad de Piura. Recuperado el 22 de Octubre de 2020, de https://pirhua.udep.edu.pe/handle/11042/3486

Virnodkar, S. P., Patil, & Jha, S. (2020). CaneSat dataset to harness convolutional neural networks for Sentinel-2 sugarcane classification. *Science Direct*. Recuperado el 16 de Octubre de 2020, de https://www.sciencedirect.com/science/article/pii/S1319157820304602

Yanfen, L., Hanxiang, W., Minh, L., & Abolghasem, S. H. (2020). Recognition of crop pests in natural settings using convolutional neural networks. *Science Direct, 169*. Obtenido de https://www.sciencedirect.com/science/article/abs/pii/S0168169919313638

ANEXOS

Anexo 1

Resolución de aprobación y vigencia de tema de tesis.

FACULTAD DE INGENIERÍA, ARQUITECTURA Y URBANISMO
RESOLUCIÓN N°2431-2020/FIAU-USS

Pimentel, 31 de diciembre de 2020

VISTO:
El Acta de reunión N° 1712-2020, remitido mediante oficio N° 0263-2020/FIAU-IS-USS de fecha 18 de diciembre de 2020 del Comité de investigación de la Escuela profesional de INGENIERÍA DE SISTEMAS, para la ejecución de la Tesis: **"DESARROLLO DE UN MÉTODO DE IDENTIFICACIÓN DE PERSONAS MEDIANTE EL PROCESAMIENTO DE IMÁGENES DIGITALES DE LA IMPRESIÓN PALMARIA"**, presentado por **CESPEDES ORDOÑEZ BARNY,** del Programa de estudios de **INGENIERÍA DE SISTEMAS**, y;

CONSIDERANDO:

Que, de conformidad con la Ley Universitaria N° 30220 en su artículo 48º sisque a letra dice: "La investigación constituye una función esencial y obligatoria de la universidad, que la fomenta y realiza, respondiendo a través de la producción de conocimiento y desarrollo de tecnologías a las necesidades de la sociedad, con especial énfasis en la realidad nacional. Los docentes, estudiantes y graduados participan en la actividad investigadora en su propia institución o en redes de investigación nacional o internacional, creadas por las instituciones universitarias públicas o privadas.";

Que, de conformidad con el Reglamento de grados y títulos en su artículo 21° señala: "Los temas de trabajo de investigación, trabajo académico y *tesis* son *aprobados por el Comité de investigación* y derivados a la facultad o Escuela de Posgrado, según corresponda, para la emisión de la resolución respectiva. El *periodo de vigencia de los mismos será de dos años*, a partir de su aprobación. En caso un tema perdiera vigencia, el Comité de Investigación evaluará la ampliación de la misma.

Que, de conformidad con el Reglamento de grados y títulos en su artículo 24º señala: La tesis es un estudio que debe denotar rigurosidad metodológica, originalidad, relevancia social, utilidad teórica y/o práctica en el ámbito de la escuela profesional. Para el grado de doctor se requiere una tesis de máxima rigurosidad académica y de carácter original. Es individual para la obtención de un grado; *es individual o en pares para obtener un título profesional*. Asimismo, en su artículo 25° señala: "El tema debe responder a alguna de las líneas de investigación institucionales de la USS S.A.C.".

Que, acorde a documento de vistos, el Comité de investigación de la Escuela profesional de INGENIERÍA DE SISTEMAS, se indica entre los acuerdos la aprobación del tema de Tesis: "DESARROLLO DE UN MÉTODO DE IDENTIFICACIÓN DE PERSONAS MEDIANTE EL PROCESAMIENTO DE IMÁGENES DIGITALES DE LA IMPRESIÓN PALMARIA" de la línea de investigación de INFRAESTRUCTURA, TECNOLOGÍA Y MEDIO AMBIENTE, a cargo de CESPEDES ORDOÑEZ BARNY en condición de estudiante, del Programa de estudios de INGENIERÍA DE SISTEMAS.

Estando a lo expuesto, y en uso de las atribuciones conferidas y de conformidad con las normas y reglamentos vigentes;

SE RESUELVE:

ARTÍCULO 1°: APROBAR, el tema de Tesis **"DESARROLLO DE UN MÉTODO DE IDENTIFICACIÓN DE PERSONAS MEDIANTE EL PROCESAMIENTO DE IMÁGENES DIGITALES DE LA IMPRESIÓN PALMARIA",** perteneciente a la línea de investigación de **INFRAESTRUCTURA, TECNOLOGÍA Y MEDIO AMBIENTE**, a cargo de **CESPEDES ORDOÑEZ BARNY,** del Programa de estudios de **INGENIERÍA DE SISTEMAS.**

ARTÍCULO 2°: ESTABLECER, que la inscripción del Tema de la Tesis se realice a partir de emitida la presente resolución y tendrá una vigencia de dos (02) años.

ARTÍCULO 3°: DEJAR SIN EFECTO, toda Resolución emitida por la Facultad que se oponga a la presente Resolución.

REGÍSTRESE, COMUNÍQUESE Y ARCHÍVESE

Dr. Mario Fernando Ramos Monzol
Decano - Facultad de Ingeniería, Arquitectura y Urbanismo
UNIVERSIDAD SEÑOR DE SIPÁN SAC.

MBA. María Noelia Dulce Rivera
Secretaria Académica / Facultad de Ingeniería, Arquitectura y Urbanismo
UNIVERSIDAD SEÑOR DE SIPÁN SAC.

Cc: Interesado, Archivo

Anexo 2

Protocolo de Bioseguridad y desinfección

PROTOCOLO DE BIOSEGURIDAD PARA LA RECOLECCIÓN DE DATOS DE IMPRESIONES PALMARIAS EN TIEMPO DE PANDEMIA COVID-19

1. DEFINICIÓN:

Es la agrupación de reglas y secuencia de pasos a seguir para proteger la salud del investigador y del participante, frente a factores de riesgo biológicos, químicos y físicos a los que está expuesto en el desempeño de sus actividades educativas con la finalidad de evitar el contagio y propagación del coronavirus SARS- CoV-2.

2. OBJETIVOS:
   - Evitar la propagación y transmisión del coronavirus SARS-CoV-2.
   - Promover la prevención y control de la pandemia.
   - Protección de la salud de todos los colaboradores durante la investigación
   - Obtener la información necesaria para el estudio de investigación.

3. RECURSOS:

a. HUMANOS:

Estudiante investigador.

b. MATERIALES:

- guantes de látex
- agua
- alcohol medicinal
- mascarilla.
- papel toalla
- mesa
- equipo escáner

4. RESPONSABLE: Barny Napoleón Céspedes Ordóñez

5. PROCEDIMIENTO:

N°	DESCRIPCIÓN DE LA ACTIVIDAD	FUNDAMENTO CIENTÍFICO
1.	Uso de EPP (mascarilla, guantes, visor)	Para minimizar los riesgos de contagio frente a agentes patógenos, con el fin de prevenir y proteger a los participantes o individuos.
2	Explicar al participante el procedimiento	Para brindar confianza y la participación
3	Aplicar alcohol medicinal en manos del participante.	Su poder bactericida es muy elevado, el alcohol de 70º elimina un 90% de las bacterias.
4	Pedir al participante posicionar la palma de la mano en el equipo escáner en las marcas preestablecidas	Un escáner de ordenador se utiliza para reproducir, utilizando la luz, imágenes impresas o documentos a formato digital.

5	Verificar la impresión palmaria	La contrastación de archivos y pruebas permitirán detectar a tiempo y modificar los errores de manera eficaz y eficiente.
6	Desinfección del área	La desinfección reúne una serie de pasos tipo químico que elimina o erradica los microorganismos sin discriminación, tales como las bacterias, virus y protozoos evitando el desarrollo de microorganismos patógenos en fase vegetativa que se encuentren en objetos inactivos.

Anexo 3

Instalación de IDE anaconda

Anexo 4

Código de implementación de Red neuronal VGG16

```
# -*- coding: utf-8 -*-
# ------------------- CGG16 ----------------------- #
import logging
from os import environ
import warnings

warnings.filterwarnings('ignore', category=FutureWarning)
environ['TF_CPP_MIN_LOG_LEVEL'] = '3'  # FATAL
logging.getLogger('tensorflow').setLevel(logging.FATAL)
# ------------------------------------------------------------------
--- #
from keras.models import model_from_json
import efficientnet.keras
import numpy as np
import json
from time import time
from sklearn.metrics import classification_report, confusion_matrix
from keras.preprocessing.image import ImageDataGenerator


def build_model(path_to_model_arch, path_to_model_weights, path_to_m
odel_labels, target_size):
    st = time()

    # CONSTRUIR EL MODELO, CARGAR PESOS Y ETIQUETAS
    print("LOADING MODEL...")
    with open(path_to_model_arch, "r") as model_json:  # ABRE EL JSO
N CON LA ARQUITECTURA DEL MODELO Y LO CONSTRUYE
        model = model_from_json(model_json.read())
    model.load_weights(path_to_model_weights)  # CARGA LOS PESOS GUA
RDADOS EN EL ENTRENAMIENTO

    with open(path_to_model_labels, "r") as json_file:  # ABRE LA LI
STA DE ETIQUETAS PARA CONVERTIR OUTPUT DEL MODELO
        labels = json.load(json_file)  # EN LOS NOMBRE QUE TIENEN LA
S ETIQUETAS DEL DATASET

    # PREDECIR CON DATA RANDOM PARA 'PRECALENTAR' EL MODELO (SIEMPRE
 LA PRIMER PREDICCION ES LENTA)
    model.predict(np.random.rand(1, *target_size, 3)).astype(np.floa
t32)
```

Anexo 5

Código completo de implementación de Red neuronal MobileNet121

```
# -*- coding: utf-8 -*-
# ------------------- IGNORAR WARNINGS MOLESTOS --------------------
--- #
import logging
from os import environ
import warnings

warnings.filterwarnings('ignore', category=FutureWarning)
environ['TF_CPP_MIN_LOG_LEVEL'] = '3'  # FATAL
logging.getLogger('tensorflow').setLevel(logging.FATAL)
# ------------------------------------------------------------------
--- #
from keras.models import model_from_json
import efficientnet.keras
import numpy as np
import json
from time import time
from sklearn.metrics import classification_report, confusion_matrix
from keras.preprocessing.image import ImageDataGenerator


def build_model(path_to_model_arch, path_to_model_weights, path_to_m
odel_labels, target_size):
    st = time()

    # CONSTRUIR EL MODELO, CARGAR PESOS Y ETIQUETAS
    print("LOADING MODEL...")
    with open(path_to_model_arch, "r") as model_json:  # ABRE EL JSO
N CON LA ARQUITECTURA DEL MODELO Y LO CONSTRUYE
        model = model_from_json(model_json.read())
    model.load_weights(path_to_model_weights)  # CARGA LOS PESOS GUA
RDADOS EN EL ENTRENAMIENTO

    with open(path_to_model_labels, "r") as json_file:  # ABRE LA LI
STA DE ETIQUETAS PARA CONVERTIR OUTPUT DEL MODELO
        labels = json.load(json_file)  # EN LOS NOMBRE QUE TIENEN LA
S ETIQUETAS DEL DATASET

    # PREDECIR CON DATA RANDOM PARA 'PRECALENTAR' EL MODELO (SIEMPRE
 LA PRIMER PREDICCION ES LENTA)
    model.predict(np.random.rand(1, *target_size, 3)).astype(np.floa
t32)
```

Anexo 6

Código completo de implementación de Red neuronal VGG19

```
# -*- coding: utf-8 -*-
# ------------------- VGG19 ----------------------- #
import logging
from os import environ
import warnings

warnings.filterwarnings('ignore', category=FutureWarning)
environ['TF_CPP_MIN_LOG_LEVEL'] = '3'  # FATAL
logging.getLogger('tensorflow').setLevel(logging.FATAL)
# ------------------------------------------------------------------
--- #
from keras.models import model_from_json
import efficientnet.keras
import numpy as np
import json
from time import time
from sklearn.metrics import classification_report, confusion_matrix
from keras.preprocessing.image import ImageDataGenerator


def build_model(path_to_model_arch, path_to_model_weights, path_to_m
odel_labels, target_size):
    st = time()

    # CONSTRUIR EL MODELO, CARGAR PESOS Y ETIQUETAS
    print("LOADING MODEL...")
    with open(path_to_model_arch, "r") as model_json:  # ABRE EL JSO
N CON LA ARQUITECTURA DEL MODELO Y LO CONSTRUYE
        model = model_from_json(model_json.read())
    model.load_weights(path_to_model_weights)  # CARGA LOS PESOS GUA
RDADOS EN EL ENTRENAMIENTO

    with open(path_to_model_labels, "r") as json_file:  # ABRE LA LI
STA DE ETIQUETAS PARA CONVERTIR OUTPUT DEL MODELO
        labels = json.load(json_file)  # EN LOS NOMBRE QUE TIENEN LA
S ETIQUETAS DEL DATASET

    # PREDECIR CON DATA RANDOM PARA 'PRECALENTAR' EL MODELO (SIEMPRE
 LA PRIMER PREDICCION ES LENTA)
    model.predict(np.random.rand(1, *target_size, 3)).astype(np.floa
t32)
```

Anexo 7

Código completo de implementación de Red neuronal Xception

```
# -*- coding: utf-8 -*-
# ----------- IGNORAR WARNINGS MOLESTOS ---------------------- #
import logging
from os import environ
import warnings

warnings.filterwarnings('ignore', category=FutureWarning)
environ['TF_CPP_MIN_LOG_LEVEL'] = '3'  # FATAL
logging.getLogger('tensorflow').setLevel(logging.FATAL)
# ------------------------------------------------------------- #
from keras.models import model_from_json
import efficientnet.keras
import numpy as np
import json
from time import time
from sklearn.metrics import classification_report, confusion_matrix
from keras.preprocessing.image import ImageDataGenerator
def build_model(path_to_model_arch, path_to_model_weights, path_to_m
odel_labels, target_size):
    st = time()
    print("LOADING MODEL...")
    with open(path_to_model_arch, "r") as model_json:  # ABRE EL JSO
N CON LA ARQUITECTURA DEL MODELO Y LO CONSTRUYE
        model = model_from_json(model_json.read())
    model.load_weights(path_to_model_weights)
    with open(path_to_model_labels, "r") as json_file:
        labels = json.load(json_file)
    model.predict(np.random.rand(1, *target_size, 3)).astype(np.floa
t32)
    et = time() - st  # TOMAR EL TIEMPO DE CONSTRUCCION DEL MODELO
    print("LOADING TIME: {:.4f} seconds".format(et))
    return model, labels
def evaluate(path_to_model_arch, path_to_model_weights, path_to_mode
l_labels, path_to_dir, target_size):
    model, labels = build_model(path_to_model_arch, path_to_model_we
ights, path_to_model_labels, target_size)

    # DEFINIMOS UN GENERADOR QUE ES EXACTAMENTE IGUAL AL UTILIZADO E
N EL ENTRENAMIENTO PARA EL SPLIT DE TESTING
    evaluation_datagen = ImageDataGenerator(
```

Anexo 8

Código completo de implementación de Red neuronal ResNet50

```
# -*- coding: utf-8 -*-
# ------------------- Restnet ----------------------- #
import logging
from os import environ
import warnings

warnings.filterwarnings('ignore', category=FutureWarning)
environ['TF_CPP_MIN_LOG_LEVEL'] = '3'  # FATAL
logging.getLogger('tensorflow').setLevel(logging.FATAL)
# ------------------------------------------------------------------
--- #
from keras.models import model_from_json
import efficientnet.keras
import numpy as np
import json
from time import time
from sklearn.metrics import classification_report, confusion_matrix
from keras.preprocessing.image import ImageDataGenerator


def build_model(path_to_model_arch, path_to_model_weights, path_to_m
odel_labels, target_size):
    st = time()

    # CONSTRUIR EL MODELO, CARGAR PESOS Y ETIQUETAS
    print("LOADING MODEL...")
    with open(path_to_model_arch, "r") as model_json:  # ABRE EL JSO
N CON LA ARQUITECTURA DEL MODELO Y LO CONSTRUYE
        model = model_from_json(model_json.read())
    model.load_weights(path_to_model_weights)  # CARGA LOS PESOS GUA
RDADOS EN EL ENTRENAMIENTO

    with open(path_to_model_labels, "r") as json_file:  # ABRE LA LI
STA DE ETIQUETAS PARA CONVERTIR OUTPUT DEL MODELO
        labels = json.load(json_file)  # EN LOS NOMBRE QUE TIENEN LA
S ETIQUETAS DEL DATASET

    # PREDECIR CON DATA RANDOM PARA 'PRECALENTAR' EL MODELO (SIEMPRE
 LA PRIMER PREDICCION ES LENTA)
    model.predict(np.random.rand(1, *target_size, 3)).astype(np.floa
```

Anexo 9

Código completo de implementación de Red neuronal DenseNet121

```
# -*- coding: utf-8 -*-
# ------------------- Densenet121 ----------------------- #
import logging
from os import environ
import warnings

warnings.filterwarnings('ignore', category=FutureWarning)
environ['TF_CPP_MIN_LOG_LEVEL'] = '3'  # FATAL
logging.getLogger('tensorflow').setLevel(logging.FATAL)
# ------------------------------------------------------------------
--- #
from keras.models import model_from_json
import efficientnet.keras
import numpy as np
import json
from time import time
from sklearn.metrics import classification_report, confusion_matrix
from keras.preprocessing.image import ImageDataGenerator


def build_model(path_to_model_arch, path_to_model_weights, path_to_m
odel_labels, target_size):
    st = time()

    # CONSTRUIR EL MODELO, CARGAR PESOS Y ETIQUETAS
    print("LOADING MODEL...")
    with open(path_to_model_arch, "r") as model_json:  # ABRE EL JSO
N CON LA ARQUITECTURA DEL MODELO Y LO CONSTRUYE
        model = model_from_json(model_json.read())
    model.load_weights(path_to_model_weights)  # CARGA LOS PESOS GUA
RDADOS EN EL ENTRENAMIENTO

    with open(path_to_model_labels, "r") as json_file:  # ABRE LA LI
STA DE ETIQUETAS PARA CONVERTIR OUTPUT DEL MODELO
        labels = json.load(json_file)  # EN LOS NOMBRE QUE TIENEN LA
S ETIQUETAS DEL DATASET

    # PREDECIR CON DATA RANDOM PARA 'PRECALENTAR' EL MODELO (SIEMPRE
 LA PRIMER PREDICCION ES LENTA)
    model.predict(np.random.rand(1, *target_size, 3)).astype(np.floa
```

Anexo 10

Carta de respuesta de empresa

JUVAL CONSTRUCTORA INMOBILIARIA

"Año de la Universalización de la Salud"

Lima de 9 diciembre 2020

Señor:
Mag. Ing. Heber Ivan Mejia Cabrera
Director de la Escuela Profesional de la Ingeniería de Sistemas
Universidad Señor de Sipán S.A.C.

Asunto:

Recolección de Información para Investigación de Universidad

Previo saludo, en atención al requerimiento solicitado por su estudiante Barny Napoleon Cespedes Ordoñez de la Escuela Profesional de Ingeniería de Sistemas, se acepta la recolección de información para fines de elaboración de investigación de tesis, en nuestro centro de labores para el presente 2020.

Cordialmente,

Melania Morales
Supervisor de Seguridad y Salud Ocupacional