Reconocimiento facial en tiempo real orientado a video llamadas o live stream para autenticar identidades durante una audiencia legal

Trabajo De Grado Para Obtener El Título De

Ingeniero Electrónico

Universidad Santo Tomás Seccional Tunja

Julián David Pardo Morcote

Junio 2020.

EXONERACIÓN DE RESPONSABILIDADES

La Universidad Santo Tomás y los autores no se hacen responsables del uso indebido de este proyecto, teniendo en cuenta que es necesario la manipulación de datos personales y cuyo manejo sin consentimiento o aprobación previa por parte de los usuarios, puede incurrir en delitos informáticos o violación a la privacidad de datos biométricos personales, derecho que esta cobijado por ley (Ley de protección de datos personales), pero que presenta algunas excepciones según la entidad que los manipule.

DEDICATORIA

En primer Lugar, dedicarle este trabajo a Dios, a mi madre Hilda Leonor Morcote Coronado quien fue el apoyo principal e incondicional durante este largo proceso, a mis hermanas Claudia Liliana Pardo Morcote, Sonia Milena Pardo Morcote, y en general a mi núcleo familiar.

AGRADECIMIENTOS

A cada uno de los docentes de las distintas áreas como ciencias básicas, ciencias humanísticas, a los docentes de la Facultad de Ingeniería Electrónica que me guiaron en cada una de las asignaturas, a mis tutores de tesis de grado Ingeniero Daniel Rodríguez Caro, Ingeniero Camilo Pardo Beainy y a la Universidad Santo Tomás Tunja por ser mi alma mater y haberme formado como un profesional ético con cualidades humanísticas.

TABLA DE CONTENIDOS

LISTA DE ILUSTRACIONES

1 GLOSARIO

Python:

Python es un lenguaje de programación interpretado, quiere decir que se ejecuta sin ser procesado por algún compilador y puede detectar errores en tiempo real, soporta tipos de programación como programación funcional, programación imperativa y programación orientada a objetos, es un lenguaje multiplataforma disponible para sistemas operativos como lo son Windows, Linux o MAC y es de código abierto, o sea que no necesita de una licencia para poder ser utilizado. (Duque, 2020)

Librería:

En programación una librería es un conjunto de algoritmos funcionales con tareas específicas codificados en algún lenguaje de programación. (zator.com, 2020)

Algoritmo:

Un algoritmo es la secuencia de ejecución de tareas de manera ordenada para obtener la solución de un problema. (tecnologia informatica, 2020)

Interfaz Gráfica:

Una interfaz gráfica es un programa informático que gestiona la interacción entre Usuario y máquina relacionándolos a través de un conjunto de objetos gráficos que representan información o que permiten interactuar entre humano y hardware. (Guay, 2010)

Base de Datos:

Es un conjunto de datos contextos o características similares almacenados de manera sistemática que pueden ser consultados de manera rápida. (ADIEGO RODRIGUEZ, 2007)

Software:

Es un sistema informático que posee algoritmos de cómputo qué le permiten al sistema operativo realizar distintas tareas. (significados.com, 2020)

Inteligencia artificial:

Es la capacidad de una máquina para interpretar datos de su entorno, aprender de estos datos y a partir de estos imitar las funciones cognitivas de un humano para tomar decisiones y hacer predicciones para realizar tareas de manera concreta. (Russell & Norvig, 2009)

Red neuronal:

Una red neuronal es un modelo informático computacional inspirado en el funcionamiento del cerebro, la red está conformada por clasificadores (neuronas), que forman unas capas de entrada encargadas de recibir datos, capas intermedias encargadas de procesar, clasificar la información que recibe y una capa de salida con la predicción de la red neuronal, producto del procesamiento de la información en sus capas intermedias. Este sistema está en la capacidad de aprender autónomamente y a partir de este aprendizaje poder predecir. (Inovation, 2019)

Visión Artificial:

Es un campo de la inteligencia artificial que combina técnicas tanto de hardware como de software para lograr capturar imágenes con el fin de analizar y procesar imágenes, básicamente la

visión artificial busca simular la visión humana mediante técnicas computacionales. (contabal, 2016)

Red LAN:

Una red LAN son equipos de cómputo conectados entre sí a través de un router, que permite la transición de datos y comunicación entre equipos a través de un protocolo de comunicación universal de redes TCP/IP (direccionamiento de paquetes de datos por dirección IP). (NetCloud, 2019)

2 RESUMEN

En este libro se encuentra la descripción del diseño y desarrollo de la aplicación para reconocimiento facial en tiempo real, se documenta todo el desarrollo de esta herramienta con alta calidad investigativa y buenos resultados en las pruebas realizadas, con el fin de aportar una solución a la problemática de suplantación de identidades, estableciendo como objetivo crear una herramienta informática para autenticar identidades durante video llamadas, que se construyó gracias a la herramienta DLIB una librería que ayuda a detectar objetos, en este caso el rostro de una persona. Se implementó el método ingenieril con enfoque aplicativo ya que no cualifica ni cuantifica variables y en conclusión se obtiene un software especializado en reconocimiento biométrico facial de alta fiabilidad, convirtiendo a este libro en una fuente de consulta seria y organizada que aporta conocimiento a la academia y a los ingenieros electrónicos que deseen realizar una investigación relacionada con reconocimiento biométrico facial.

3 INTRODUCCIÓN

El reconocimiento facial está siendo implementado en muchas aplicaciones de seguridad principalmente para confirmar o autenticar identidades y de esta manera evitar la suplantación de personas. Tecnológicamente se está implementando inteligencia artificial una técnica que actualmente permite identificar una persona con una exactitud del 99 %, es decir una tecnología con una alta fiabilidad de autenticidad para identidades.

Debido a la situación actual donde los ciudadanos están sometidos a un aislamiento social por causa de la aparición de un virus mortal del año en curso, se ha optado por dar un manejo laboral de manera virtual, lo cual conlleva a tener que realizar reuniones o trabajos por video llamada y para el sector judicial es fundamental contar con la veracidad de identidades por lo que se ideó una manera de identificar una persona durante el transcurso de una video llamada.

Python es un lenguaje de programación bastante flexible y muy amplio que hoy en día está teniendo una acogida bastante grande por parte de los desarrolladores y científicos de datos, además de ser un lenguaje de programación sencilla apto para realizar operaciones lógicas complejas con el fin de crear aplicaciones optimas que satisfagan la solución a problemas informáticos y que permitan el avance evolutivo de tecnología enfocada al software. Por esta razón la herramienta de reconocimiento facial para video llamadas fue programada aplicando algoritmos de inteligencia artificial y visión por computador compatibles con la plataforma Python.

4 JUSTIFICACIÓN

Colombia es un país con altos índices de corrupción donde la suplantación de identidad es frecuente, lo cual genera un problema social que afecta a toda su población ya sea de manera directa o indirecta, posicionándolo internacionalmente como uno de los países con mayor índice de corrupción; por esta razón se enfoca el campo investigativo a la creación y puesta en marcha de un sistema que permita reconocer una persona biométricamente para combatir acciones fraudulentas, además de brindar seguridad, confianza y transparencia en procesos legales, específicamente durante las audiencias virtuales desarrolladas en la corte consejo de estado del palacio de justicia.

En cuanto a su aporte en la parte social, debido a la emergencia de salubridad que atraviesa el planeta actualmente por causa del virus COVID-19, la implementación de esta tecnología es significativa pues no se requiere de contacto físico para identificar una persona biométricamente, como si pasa con otras herramientas de reconocimiento las cuales necesitan el contacto físico en algún tipo de superficie (como puede ser la lectura de huella digital), es así como al desarrollar el reconocimiento facial se contribuye a evitar la propagación de este virus o de cualquier otro con características similares al coronavirus.

El impacto medioambiental es mínimo ya que al tratarse de un sistema informático este no contamina y tampoco genera desecho alguno, sin embargo, necesita de energía eléctrica para funcionar por lo cual el impacto ambiental que precede de la generación energética influye en segundo plano a la ejecución del proyecto.

En la parte económica no perjudica ni beneficia a ningún sector en específico por lo que el impacto económico es mínimo, ya que este proyecto investigativo pretende implementar un sistema de seguridad informático que no requiere de una significativa inversión. Por otra parte, se encuentra

abierta la posibilidad de crear nuevos sistemas de reconocimiento facial derivados de este proyecto, que a futuro puedan generar interés de inversión por parte de empresas o particulares, y de esta manera generar un impacto económico positivo.

5 PLANTEAMIENTO DEL PROBLEMA

5.1 Formulación de preguntas

- ¿Por qué es importante implementar el sistema de reconocimiento facial en un proceso jurídico?

- ¿Cómo puede una computadora reconocer facialmente una persona y autenticar su identidad?

5.2 Definición del problema

La suplantación de identidad es un acto fraudulento y deshonesto que afecta a la sociedad y que puede presentarse en cualquier eventualidad cotidiana, desencadenando en engaño a la veracidad y transparencia de cualquier acto civil, específicamente a quienes buscan impartir justicia en un proceso legal judicial.

5.3 Delimitación del problema

El sistema biométrico facial que se realizará en este proyecto podrá ser usado para autenticar la identidad de personas que se involucren en un proceso legal ya sea un juez, fiscal, abogados, testigos demandante o demandado, cuya intervención sea mediante una video llamada o live stream, para las audiencias virtuales realizadas en la corte consejo de estado del palacio de justicia de la ciudad de Bogotá.

6 OBJETIVOS

6.1 Objetivo general

Diseñar y crear un sistema informático de reconocimiento facial en tiempo real orientado a videollamadas para autenticar identidades durante una audiencia legal.

6.2 Objetivos específicos

- Programar el software de reconocimiento facial, tanto interfaz como algoritmo, en la plataforma de lenguaje Python implementado librerías de visión por computador e inteligencia artificial.

- Crear una interfaz sencilla cuya función sea identificar y registrar información del reconocimiento facial de manera gráfica.

- Evaluar la efectividad de este software realizando pruebas de reconocimiento facial con distintas personas que voluntariamente se sometan y que no sean necesariamente funcionarios de la corte consejo de estado.

7 MARCO TEORICO

Los inicios del reconocimiento facial datan para el año 1960 cuando Woodrow Wilson Bledsoe ideó un sistema con el que pudo clarificar los distintos rasgos del rostro de una persona mediante el uso de la tabla RAND. El sistema implementaba un lápiz óptico para ubicar los ojos boca y nariz, pero era un sistema que debía usarse manualmente.

Durante diez años se fue mejorando esta técnica para hacerla más precisa, luego para 1980 se hace un aporte a la técnica aplicando algebra lineal para conseguir mejoras. Para el año 1991 los desarrolladores Pentland y Turk desarrollaron un modelo de detección capaz de identificar el rostro de una persona de manera automática. Luego en el año 2001 se crea el Viola-Jones Object Detection Framework, algoritmo para la detección de rostros y finalmente en la última década se implementa la inteligencia artificial en forma de Convolutional Neural Networks, un sistema de aprendizaje profundo automático capaz de detectar e identificar el rostro de una persona ya sea a través de una imagen video o video cámara. (PADigital, 2018)

El sistema informático de reconocimiento facial implementado para este trabajo opera con cuatro principales librerías fundamentales para el proceso de identificación como lo son PyAutogui, Opencv, dlib y face_recognition, a continuación se define cada una de estas y de la manera como interviene durante el proceso de identificación.

7.1 PyAutogui

Esta es una librería que contiene varias funciones como el control de la posición del mouse, control de los botones de interacción del mouse, cajas con mensajes de alertas, control del teclado y captura de la pantalla del equipo de cómputo. (Sphinx, 2019)

Capturar la pantalla es la función principal que el sistema de reconocimiento utiliza para capturar la ventana de la videollamada y almacenarla temporalmente en un formato de imagen, que posteriormente será procesada por las demás librerías para encontrar caras y proceder a identificarlas.

7.2 Opencv

Esta es una librería de visión artificial que permite la captura y procesamiento de imágenes desde una video cámara, compara imágenes, realiza seguimiento de objetos en movimiento, detecta rasgos faciales y hace modelamiento 3D entre otras funciones. (Opencv.org, 2020)

Esta librería interviene en la captura de imágenes desde la cámara web del equipo de cómputo para la creación en la base de datos de una cara nueva que el sistema va a identificar en un futuro, también adiciona de manera gráfica la información de la hora y fecha en el momento en que se realiza el reconocimiento facial.

7.3 Dlib

Es una librería de aprendizaje automático moderno (machine learning) creada en lenguaje C++ lo cual permite adaptarse y utilizarse en muchos lenguajes de programación ya que son algoritmos de código abierto, entre esos Python. Esta es la librería quizás más importante en cuanto a reconocimiento facial se refiere, porque ha desarrollado algoritmos para la detección de objetos entre otros los rasgos biométricos faciales. Dicha librería está en la capacidad de detectar un rostro en una imagen, detectar sus bordes y distribuir sesenta y ocho landmarks (puntos de referencia), en los bordes detectados.

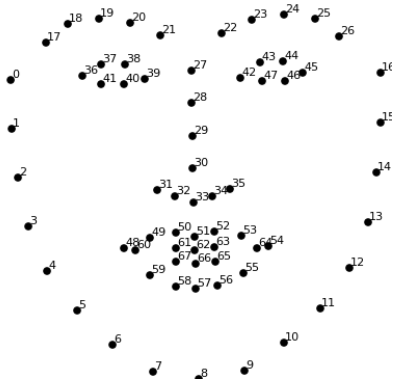


Ilustración 1 Puntos de referencia que se dibujan en los bordes de cara detectadas

Fuente: ResearchGate, Amos 2016

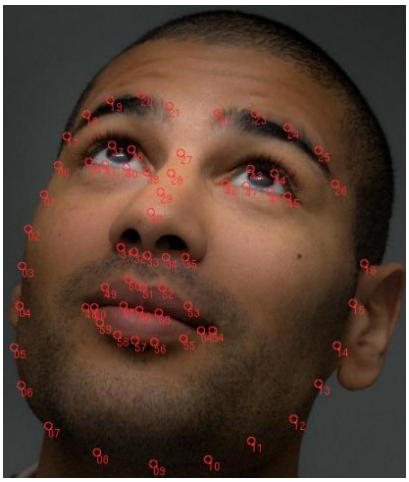


Ilustración 2 Cara con bordes detectados y landmarks distribuidas en los bordes.

Fuente: blog.dlib.net, King 2017

Mediante un algoritmo euclidiano mide la distancia entre cada landmark y luego compara estas mediciones con las de una cara previamente codificada, si hay similitudes entre las distancias el sistema predice que es una persona reconocida, pero si los valores no coinciden se debe a una persona desconocida. La red neuronal convolucional está compuesta por 29 capas y fue entrenada

con tres millones de caras lo cual garantiza una precisión del 99.38 % al momento de detectar e identificar un rostro.

A continuación se muestra la arquitectura general para una red neuronal convolucional y los procesos o clasificaciones dentro de las capas de convolución, filtros kernel, Subsampling, convoluciones, fully conected, softmax y clasificación de salida one hot encode.

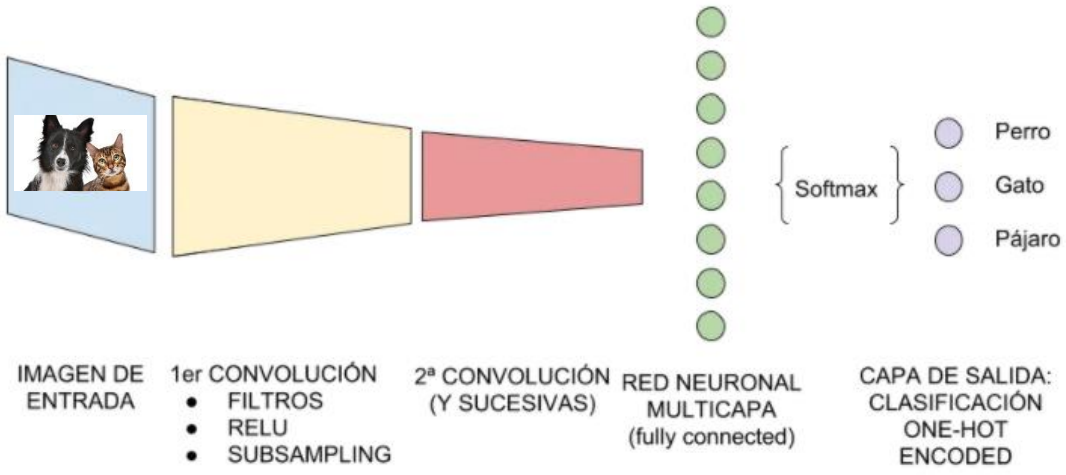


Ilustración 3 Arquitectura de una red neuronal convolucional

Fuente: aprende machine learning, Na8 2019

El proceso general de una red neuronal convolucional es tener una imagen de entrada (matriz de pixeles), aplicar filtros kernel, obtener un mapeo de características (feature mapping), aplicar max pooling y finalmente obtener salida de convolución. Es aquí donde termina el proceso de convolución.

Luego de las convoluciones se hace fully conected, (conectar a una red neuronal tradicional), seguido de la función softmax, y finalmente la capa de clasificación one-hot encode (clasificador).

Las primeras capas convolucionales pueden detectar líneas o curvas, pero a medida que aumenta el número de capas de convolución, la red neuronal puede reconocer formas más complejas. (Na8, 2019)

Kernel: Es un filtro dentro de la convolución que consiste en realizar una operación matricial (producto escalar), entre los pixeles de imagen de entrada con una pequeña matriz llamada kernel, obteniendo nuevas imágenes de salida con mapeo de características (feature mapping). Este filtro captura las características más relevantes de la imagen de entrada y las dibuja en las nuevas imágenes de salida (feature mapping).

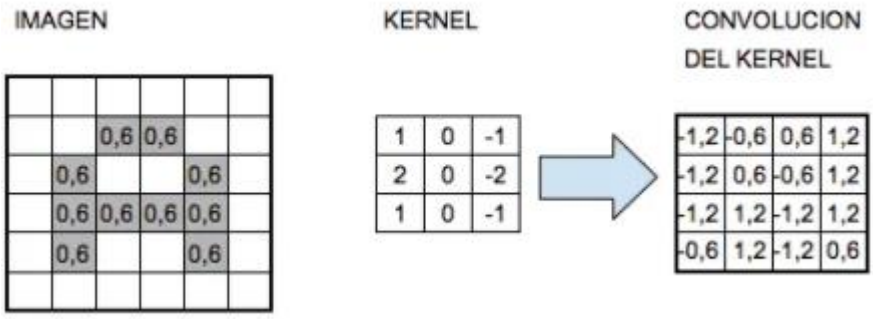


Ilustración 4 filtrado de kernel

Fuente: aprende machine learning, Na8 2019

Subsampling: Es un proceso en la última etapa de la convolución que consiste en la reducción del tamaño de las imágenes obtenidas en feature mapping con el fin de aumentar la velocidad de clasificación y disminuir procesamiento. Para este proceso se utiliza la función conocida como max pooling.

Max pooling: Esta función captura una porción de pixeles y los transforma en un único pixel con el valor más alto de los pixeles de la sección tomada por eso el nombre de “Max”.

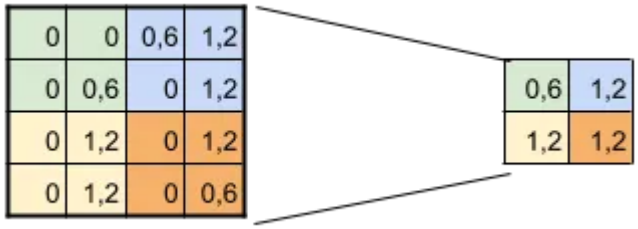


Ilustración 5 Subsampling Max pooling 2x2, se reduce la información a la mitad.

Fuente: aprende machine learning, Na8 2019

Fully conected: Esta función toma las salidas de las capas anteriores y las "aplana", quiere decir que las transforma en una nueva imagen a partir de todas las imágenes obtenidas en Subsampling para que sean la entrada de la próxima convolución o en caso dado de sea la última capa prepararla para aplicar softmax.

Softmax: Es la última capa que contiene las neuronas de clasificación, en caso de clasificar perro y gatos se tienen dos neuronas, en caso de datos numéricos se tienen diez neuronas y si se clasifica aviones, coches y barcos se tienen tres neuronas, y así sucesivamente con los objetos a clasificar.

Esta función es la encargada de entregar la información de predicción entre 0 y 1 en un formato conocido como **one-hot encode** para el caso de clasificar perros y gatos seria [1,0] o [0,1]. La salida one-hot encode [0.2 0.8], indica que hay un 20% de probabilidad que sea un perro y un 80% que sea gato.

Esta es la explicación en términos generales de cómo trabaja un red neuronal convolucional, para clasificar objetos en imágenes.

7.4 Face-recognition

Esta librería fue creada con el fin de programar la librería dlib de manera sencilla en Python, quiere decir que implementa comandos mucho más simples que invocan todo el proceso de reconocimiento desde dlib y hace que con tan solo cinco funciones puede realizar reconocimiento facial de manera rápida. (Geitgey, 2017)

Más adelante en el capítulo tres de la sección de resultados dentro de este documento se explica detalladamente el proceso de cada una de las funciones implementadas en el código del algoritmo.

8 METODOLOGÍA

Durante el desarrollo de este proyecto se pone en práctica el modelo ingenieril con un diseño de corte transversal ya que las pruebas fueron tomadas en un solo momento, con el fin de poder entregar resultados confiables, verídicos y con alta calidad durante todas las etapas de ejecución. Inicialmente se realiza un estudio relacionado con la metodología de la investigación, encontrando a Sampieri como uno de los mayores referentes de los procesos investigativos tanto de diseño y enfoque. Es Roberto Sampieri quien en su libro metodología de la investigación explica de manera detallada los alcances investigativos como lo son: exploratorio, descriptivo, correlacional y explicativo, los cuales tienen un enfoque cualitativo o cuantitativo, (Sampieri, 2014), sin embargo, para el proyecto de reconocimiento facial se busca dar un enfoque aplicado, ya que no se pretende cualificar o cuantificar variables, por esta razón se analizan otros posibles métodos y finalmente se elige el método ingenieril el cual tiene un tipo de estudio que mejor se acopla a proyectos aplicativos y que busca dar solución a problemáticas específicas.

8.1 **Tipo de estudio**

8.1.1 Método ingenieril

Este método se basa en la toma de decisiones para desarrollar materiales, productos o procesos que necesiten satisfacer una necesidad. La definición conceptual exacta de este método es: “estrategia para producir el cambio con los recursos disponibles en una situación deficientemente entendida o incierta”. (Wesly, 1994)

Por esta razón se desea implementar el método ingenieril, ya que este método sugiere seguir una serie de pasos para el diseño y ejecución de un proyecto como lo son:

- Partir de la necesidad de dar solución a un problema.
- Realizar un estudio de factibilidad y viabilidad.
- Buscar información relacionada con la temática del problema y de sus posibles soluciones.
- Delimitar la solución del proyecto.
- Desarrollar varios conceptos y diseños alternos.
- Seleccionar el diseño más promisorio.
- Optimizar el diseño, implementarlo y finalmente evaluarlo

Para explicar de mejor manera el desarrollo y ejecución del proyecto de "Reconocimiento facial en tiempo real orientado a video llamadas o live stream para autenticar identidades durante una audiencia legal" se describen de manera específica las distintas etapas que se realizarán para conseguir cumplir con el objetivo general y los objetivos específicos propuestos en este proyecto.

8.1.2 Etapa investigativa

Se procede a iniciar una investigación de artículos y proyectos de reconocimiento facial que ya fueron desarrollados e implementados con el fin de conocer el estado del arte actual, además se realiza una investigación es la base de datos SCOPUS para conocer el avance y el interés con el que los investigadores a nivel mundial están trabajando con el tema relacionado de reconocimiento facial, visión artificial y deep learning, más adelante en la sección de resultados se puede encontrar de manera más detallada los índices de investigación enfocados al tema de detección biométrica

facial; dentro de esta investigación se realiza el estudio del conjunto de herramientas necesarias para poder detectar un rostro, reconocer e identificar una persona en tiempo real.

Esta investigación incluye el estudio y el aprendizaje del lenguaje de programación que se usará para el diseño, desarrollo y ensamblaje del algoritmo de reconocimiento facial.

8.1.3 Etapa de diseño

Es en esta etapa donde se aplicarán todos los conceptos y conocimientos relacionados con el reconocimiento facial moderno, que incluye visión por computador, tratamiento de imágenes, inteligencia artificial y lenguaje programación, adquiridos durante la etapa de investigación. Además, durante esta etapa se procede a diseñar varios algoritmos e interfaces de usuario, con el fin de desarrollar ideas para dar un mejor enfoque al diseño final.

8.1.4 Etapa de ejecución y evaluación

Al tratarse de un sistema que será implementado por primera vez, este será sometido a una etapa de prueba donde básicamente se ejecutará y se evaluará. Para esta etapa el sistema informático estará terminado y se dará inicio a la implementación de la herramienta de reconocimiento facial, además se realizará una evaluación para determinar si este cumple y satisface los requerimientos del problema.

Para evaluar este sistema se realizarán pruebas de reconocimiento facial con las caras de personas que voluntariamente quieran participar de este proceso y que aprueben la manipulación de sus datos durante la etapa evaluativa, además no es necesario que estas personas sean trabajadores o funcionarios de la corte consejo de estado.

El protocolo de evaluación será el siguiente:

Se escogerán diez personas mayores de edad que deseen participar voluntariamente durante un periodo de 5 días, a quienes se les realizará un reconocimiento facial diario para generar un total de ochenta pruebas. Además, estas personas deben aportar una fotografía con un máximo de antigüedad de dos años donde su cara sea claramente visible, para poder generar una base de datos a partir de estas imágenes.

Para el reconocimiento facial diario se harán pequeñas modificaciones faciales con accesorios como lentes, tapabocas, maquillaje, cambios de luminosidad, cambios emocionales y ángulos de posición. Cabe resaltar que el sistema no puede reconocer una persona al poner su cara completamente lateral.

Finalmente cuantificar mediante una matriz de confusión (matriz que cuantifica la calidad de predicción de un sistema que implementa inteligencia artificial), cuantas veces logra el sistema informático reconocer a una persona y si este en algún momento puede confundir identidades. Todo esto se realiza con el fin de identificar bajo que variables de entorno puede o no se puede reconocer una persona facialmente.

Adicionalmente se realizó una prueba de conexión tipo cliente servidor en una red local para determinar si el software puede acceder a una base de datos que se encuentre en otro equipo de cómputo.

8.2 Variables

8.2.1 Variable dependiente

La variable dependiente de esta investigación es el reconocimiento facial, debido a que esta depende de los rasgos faciales y del reconocimiento de patrones biométricos.

8.2.2 Variable independiente

Por otro lado, la variable independiente de la investigación es la persona que es sometida al reconocimiento por parte del software, ya que los resultados no la afectarán.

8.3 Definición de hipótesis

H0 La herramienta informática de reconocimiento facial no está en la capacidad de autenticar la identidad de la persona en la video llamada.

H1 La herramienta informática de reconocimiento facial está en la capacidad de reconocer y autenticar la identidad de una persona en tiempo real durante la video llamada.

8.4 Población y muestra

Debido a la situación actual de pandemia que atraviesa el país, las pruebas se realizaron con personas que voluntariamente se sometieron al reconocimiento facial y que no hacen parte de la entidad en la cual se pretende implementar este sistema, sin embargo este proyecto va dirigido a personas que laboren con la rama judicial, que sean funcionarios de la corte consejo de estado, palacio de justicia (Bogotá distrito capital). La muestra estará representada por participantes que voluntariamente aceptaron realizar las pruebas referentes a identificación facial. Es la oficina de sistemas de la corte consejo de estado la que finalmente decidirá si implementa o no esta herramienta de reconocimiento facial. Adicional a esto será esta oficina la que determine los derechos de uso, manipulación de datos y almacenamiento de información que se adjudiquen a este proyecto, ya que el creador del software de reconocimiento facial solo desarrollará dicha herramienta, pero se excluye de cualquier responsabilidad referente a la manipulación de datos.

9 RESULTADOS

En esta sección se explica de manera detallada toda la planeación, diseño, desarrollo y resultados de todo el proyecto, por esta razón la sección de resultados se divide en tres capítulos que donde se muestra los pasos y consideraciones que se tomaron para conseguir cumplir con los objetivos del proyecto.

9.1 ETAPA 1

9.1.1 Investigación

Para dar comienzo al modelamiento del proyecto se debe tener unas bases respecto al estado del arte actual del reconocimiento facial con el fin de conocer su inicio, desarrollo y avances que ha tenido esta tecnología hasta el día de hoy, por esta razón se investiga la historia relacionada al reconocimiento biométrico facial y el desarrollo de proyectos con la misma temática para contextualizarse con dicha tecnología, para así tener un punto de referencia y comenzar a reunir información y las herramientas necesarias para la creación de un sistema informático que está en la capacidad de reconocer e identificar una persona durante una video llamada, encontrando así los siguientes artículos de proyectos desarrollados para reconocimiento facial.

A continuación se mencionan tres artículos más relevantes investigados que implementan inteligencia artificial que mejor se acoplaron al modelamiento del proyecto.

En el primer artículo con nombre “evaluación y comparación del rendimiento de software para reconocimiento facial basado en la biblioteca dlib y Opencv”, realiza una comparativa entre estas dos librerías de visión artificial donde analizan las ventajas y desventajas de cada una de estas, además de ejemplos de aplicaciones de reconocimiento de rostros basados en histogramas de imágenes, distribución de puntos de referencia, efectividad de la red neuronal convolucional y

comparación de rostros desconocidos con rostros conocidos. (Boyko, Basystiuk, & Shakhovska, 2018)

Este artículo es fundamental para realizar una introducción de las librerías necesarias para el reconocimiento facial que implementa inteligencia artificial.

El segundo artículo con nombre de "método robusto para el reconocimiento facial y el sistema de detección de emociones faciales utilizando máquinas de vectores de soporte", se realiza una investigación para poder efectuar reconocimiento facial en tiempo real de las emociones faciales, implementando algoritmos de aprendizaje automático y redes neuronales para la clasificación de diferentes clases de emociones faciales mediante un entrenamiento a través de imágenes, implementando visión artificial de código abierto y aprendizaje automático con Python. (Rajesh & Naveenkumar, 2016)

En el tercer artículo con nombre "reconocimiento facial moderno con aprendizaje profundo", se realiza una investigación donde se implementa técnicas de aprendizaje profundo y se entrena un clasificador de rostros donde se busca mejorar el reconocimiento facial debido a algunos errores en la detección de rasgos faciales por oclusiones como cambios de postura, iluminación, obstrucción y modificación de rasgos faciales, desarrollando una aplicación en Python para poder identificar un rostro en cualquier condición. (Thilaga, Khan, Jones, & Kumar, 2018)

Los dos últimos artículos resumidos anteriormente hacen énfasis en el desarrollo de una aplicación de reconocimiento facial en lenguaje Python, por tanto, se puede concluir que este lenguaje es el más viable para el desarrollo de aplicaciones de reconocimiento de rasgos faciales e identificación biométrica facial.

A continuación, se muestran una serie de gráficas tomadas del análisis de la base de datos SCOPUS, que permiten conocer la evolución histórica y actual relacionada con los trabajos y proyectos que se han desarrollado hasta septiembre de 2020 y que son producto de la investigación realizada para este trabajo.

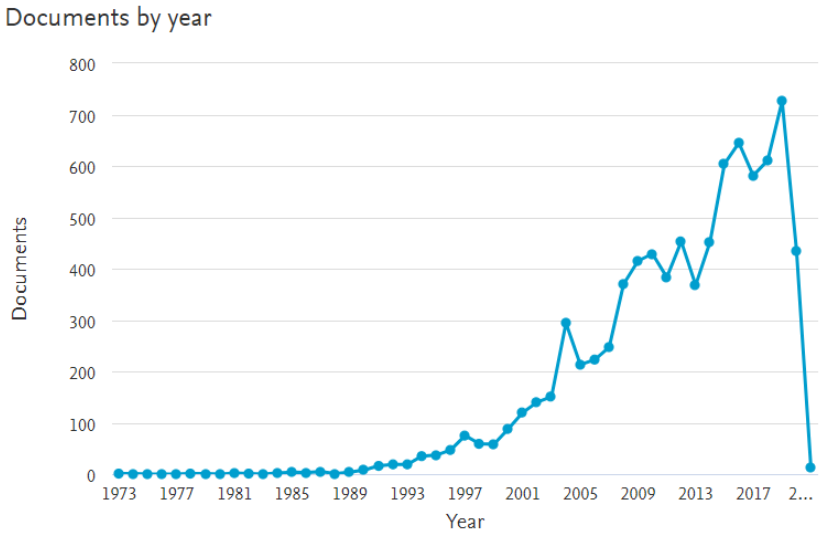


Ilustración 6 Gráfica evolutiva de desarrollo enfocado al reconocimiento facial.

Fuente: Análisis de base de datos SCOPUS.

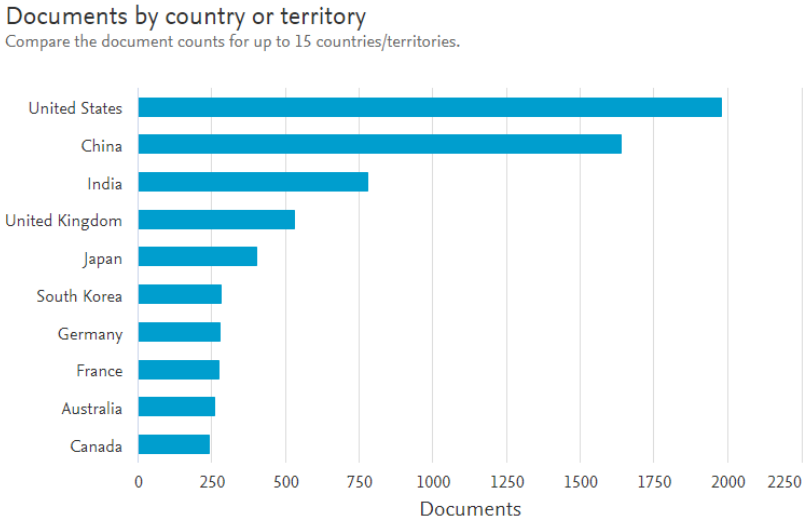


Ilustración 7 Principales países que han investigado el desarrollo de esta tecnología.

Fuente: Análisis de base de datos SCOPUS.

Seguidamente, se realiza un análisis del desarrollo de reconocimiento facial, vision por computador y deep learning, para la última década a nivel nacional en comparativa con los países pioneros en la investigación de esta tecnología, ya que se establece que una investigación con más de diez años de publicación, pierde validez debido a que se considera desactualizada.

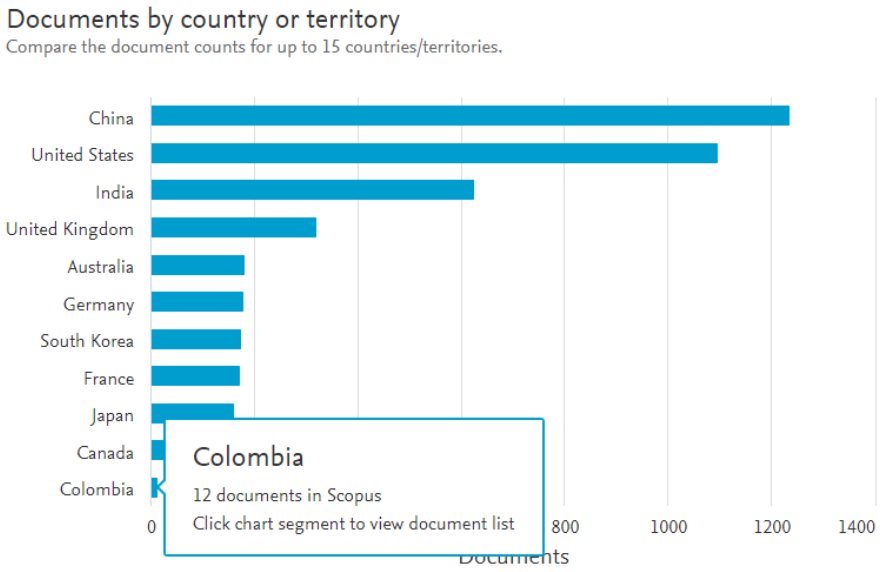


Ilustración 8 Análisis comparativo de investigaciones enfocadas a reconocimiento facial desde el año 2010 a 2019.

Fuente: Análisis de base de datos SCOPUS.

En la siguiente gráfica se puede observar cuales son las principales áreas en las que se enfoca la investigación al desarrollo de nuevas tecnologías o técnicas de reconocimiento biométrico facial.

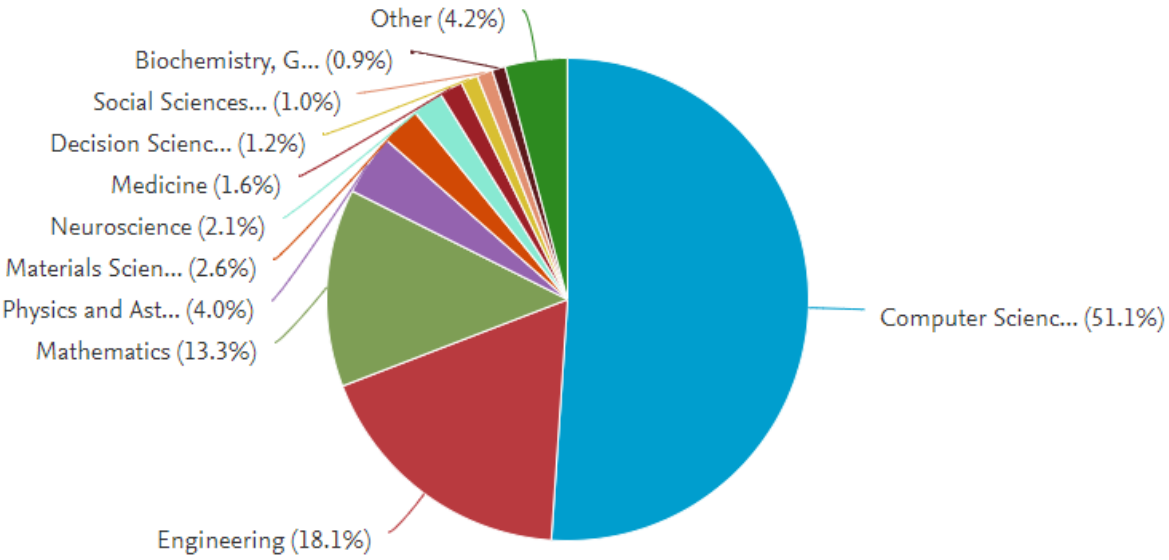


Ilustración 9 Análisis investigativo por áreas.

Fuente: Análisis de base de datos SCOPUS.

Después de la investigación de proyectos y artículos relacionados con reconocimiento facial, finalmente se reúnen todas las herramientas necesarias para comenzar a diseñar una herramienta informática capaz de identificar rasgos faciales durante una videollamada que son: Python como lenguaje de programación, Opencv y dlib como librerías de reconocimiento e inteligencia artificial. Se procede a realizar un aprendizaje profundo sobre cómo programar en lenguaje Python y cómo acoplar dichas librerías en el lenguaje mencionado. La razón por la cual se elige Python como lenguaje de programación es porque es un lenguaje de alto nivel que no necesita ser compilado y que es capaz de detectar errores en tiempo real durante su ejecución, además de ser un lenguaje empleado para el procesamiento de cálculos matemáticos complejos, data sience, tratamiento de imágenes, inteligencia artificial, etc.

A continuación, se muestra una imagen que muestra el ranking de los principales lenguajes de programación que se están empleando actualmente para el tratamiento de datos y desarrollo de aplicaciones.

Rank	Language	Type	Score
1	Python		100.0
2	Java		96.3
3	C		94.4
4	C++		87.5
5	R		81.5
6	JavaScript		79.4
7	C#		74.5
8	Matlab		70.6
9	Swift		69.1
10	Go		68.0

Ilustración 10 Ranking Top 10 lenguajes de programación más usados actualmente.

Fuente: Revista web IEEE SPECTRUM.

Este ranking posiciona a Python como el principal lenguaje de programación multiplataforma mayormente empleado, lo que ratifica la elección para el diseño y desarrollo de la aplicación de reconocimiento facial que se plantea para este trabajo de grado. Fue necesario la

realización de un curso online sobre programación en lenguaje Python de la cual se obtuvo una certificación del aprendizaje básico del lenguaje que se puede consultar en los anexos de este libro.

Durante el curso se vieron temas básicos desde tipos de variables, bucles, y condicionales hasta temas como programación orientada a objetos y herencia de funciones, así como también se aprendió a crear los ejecutables del software.

9.2 ETAPA 2

9.2.1 Diseño y desarrollo

Después de haber realizado toda la investigación pertinente, se define el lenguaje Python como lenguaje de programación para el desarrollo de los algoritmos de reconocimiento facial, para esto se implementa dos librerías encargadas del proceso de reconocimiento y machine learning, face-recognition y dlib respectivamente. Al tratarse de una video llamada se programa el algoritmo para que realice una captura de pantalla desde el borde izquierdo hasta la mitad de la pantalla, para esta tarea se necesita implementar la librería PyAutogui, la cual permite realizar un screen shot o captura de pantalla.

En el siguiente diagrama de flujo se puede ver el proceso completo del algoritmo de reconocimiento y seguido de la ilustración se describen los procesos que se realizan durante cada uno de los bloques.

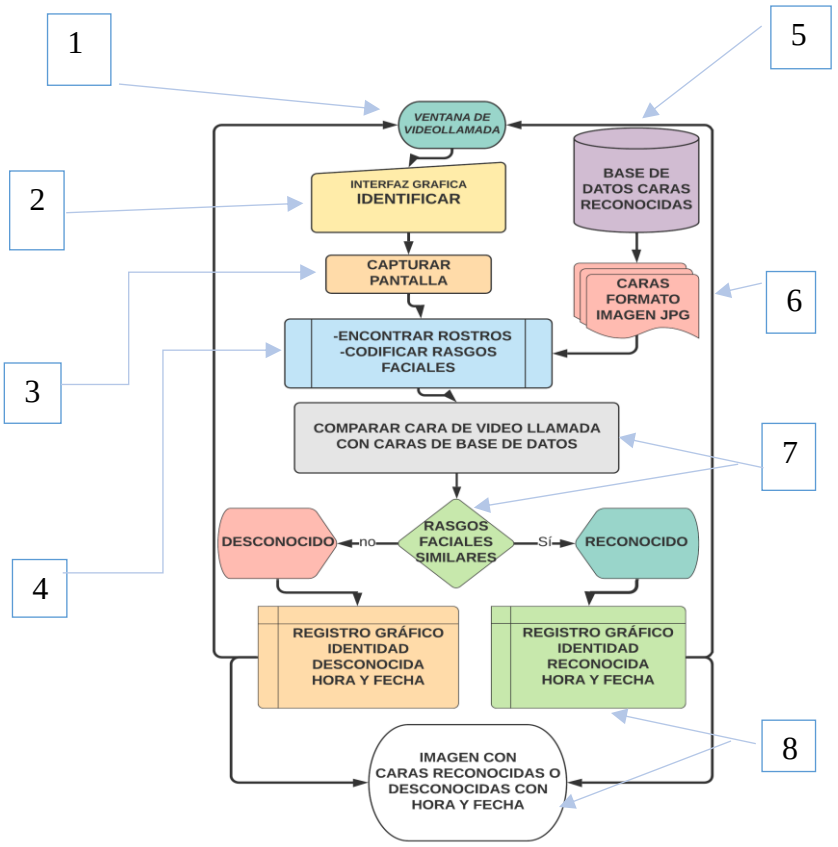


Ilustración 11 Diagrama de flujo para algoritmo de reconocimiento

Fuente: Autor.

1. La ventana debe estar ubicada en la parte izquierda de la pantalla ya que el algoritmo buscará caras solo en esta sección de la pantalla.
2. Al hacer clic en el botón de la interfaz "Identificar" se inicia el proceso de reconocimiento facial.
3. El primer paso del proceso es capturar la pantalla donde se encuentra la video llamada que debe estar ubicada en la parte izquierda.
4. En la imagen capturada de la pantalla el algoritmo se enfoca en encontrar los bordes, la red neuronal se encarga de clasificar los bordes para ubicar la cara en la imagen y luego se colocan los puntos de referencia (landmarks) sobre los bordes de la cara y estos datos se guardan en un espacio vectorial dentro del algoritmo.

5. La base de datos se compone de directorios que poseen la imagen de la cara de la persona que el algoritmo reconoce.
6. Las imágenes de caras reconocidas son accedidas mediante la red local LAN y transmitidas por el puerto ethernet con un protocolo TCP/IP.

   Estas imágenes son procesadas al igual que las imágenes entrantes de la captura de pantalla.
7. Con la información de ubicación de los puntos de referencia, interviene el algoritmo euclidiano que mide la distancia entre cada punto de referencia y si las distancias coinciden se trata de un rostro conocido, en caso contrario es una cara desconocida.
8. Se genera un registro gráfico que se guarda en una carpeta con el nombre de "Registros" y adiciona a la imagen la fecha y la hora, además que se muestra como una ventana nueva.

A continuación se observa una imagen de ejemplo del segmento de captura de pantalla donde se debe ubicar la ventana de video llamada, adicional a esto mediante la librería datetime y Opencv (cv2), se incluye la fecha y la hora para llevar un control de tiempo y poder saber en qué momento se ejecuta el programa y se realiza el reconocimiento facial, todo esto de manera gráfica, es decir que los archivos de información que se genera para los archivos de registro son archivos de imagen.

Ilustración 12 imagen generada a partir de una captura de pantalla que posteriormente será tratada para realizar reconocimiento facial

Fuente: Autor.

Seguido de esto se realiza un tratamiento a la imagen previamente generada con la librería face-recognition y Opencv, con el fin de detectar rostros en la imagen, codificar las caras para luego compararlas con las caras que están en la base de datos, o sea las caras que el algoritmo reconoce; estas caras se encuentran almacenadas en un directorio con el nombre de "CARAS/known_faces" y dentro de este directorio se encuentran subcarpetas que contienen la imagen y el nombre de la cara conocida que serán cargadas y codificadas por el algoritmo de reconocimiento.

A continuación, se muestra una parte del algoritmo que carga la base de datos, realizando la codificación de cada una de las caras conocidas y agregándolas a una lista con el nombre

correspondiente, para luego ser comparadas con la cara detectada procedente de la ventana de la video llamada y finalmente definir si es una cara reconocida o desconocida.

```
known_face_encodings = []
known_face_names = []
today = date.today()

KNOWN_FACES_DIR="CARAS/known_faces"
for name in os.listdir(KNOWN_FACES_DIR):
    for filename in os.listdir(f"{KNOWN_FACES_DIR}/{name}"):
        image = face_recognition.load_image_file(f"{KNOWN_FACES_DIR}/{name}/{filename}")
        encoding = face_recognition.face_encodings(image)[0]
        known_face_encodings.append(encoding)
        known_face_names.append(name)
```

Ilustración 13 algoritmo que realiza la carga de la base de datos de las caras conocidas.

Fuente: Autor.

El anterior código crea dos listas vacías donde se almacenan las caras codificadas y los nombres correspondientes a esas caras (Known_faces_encodings = [ ] y Known_faces_names = [ ]), luego KNOWN_FACES_DIR almacena la dirección de ubicación en disco "CARAS/known_faces". Para codificar todas las caras dentro de este directorio implementamos la librería os que permite trabajar las ubicaciones de disco y finalmente dentro de un ciclo for anidado se realiza la lectura y codificación de cada una de las caras existentes dentro de la base de datos.

Face-recognition es una librería de código abierto desarrollada en lenguaje Python que a su vez implementa dlib, la librería de reconocimiento facial más simple del mundo que incluye una red neural de tipo convolucional pre entrenada que permite detectar caras y rasgos faciales con una efectividad del 99.38%. (King, 2017)

En la sección del marco teórico de este libro, se describe en términos generales una red neuronal convolucional, sin embargo, la topología de la red neuronal de la librería dlib es una red de tipo ResNet (residual Network o red residual) de 29 capas, que ha sido entrenada con tres millones de imágenes de caras. A continuación se muestra una imagen de la arquitectura de esta red, (modelo de red ganadora del concurso NetImage challenge 2015), que presenta un error máximo del 3.6%. (Rivera, 2019)

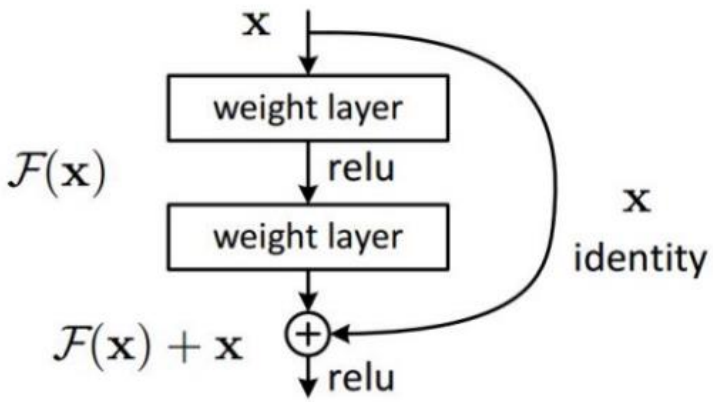


Ilustración 14 Arquitectura ResNet o red residual

Fuente: Modelos CNN en la clasificación de imágenes clásicas y modernas, Quintana 2018

Donde X es la imagen que será clasificada o la suma entre dos convoluciones previas y el resultado de dos convoluciones posteriores, es decir: X = X + F(X). Ahora F(X) es el resultado de la convolución y filtrado de dos capas de la red. En términos generales puede describirse como una red neuronal retroalimentada, característica que permite aumentar la precisión al momento de clasificar y predecir.

```python
def pantallazo():
    today = date.today()
    hora_actual = time.strftime('%H:%M:%S')
    img = ImageGrab.grab(bbox=(0, 0, 682, 650))
    img_np = np.array(img)
    pantalla = cv2.cvtColor(img_np, cv2.COLOR_BGR2RGB)
    hora = datetime.now()
    cv2.rectangle(pantalla,(0,550),(700,700),(0,0,0),cv2.FILLED)
    cv2.putText(pantalla,"FECHA:  "+str(today),(30,590),cv2.FONT_HERSHEY_SIMPLEX,1.0,(14, 201, 255),2)
    cv2.putText(pantalla,"HORA:   "+hora_actual,(30,620),cv2.FONT_HERSHEY_SIMPLEX,1.0,(14, 201, 255),2)
    #print("tomando pantalla")
    cv2.imwrite("captura.jpg", pantalla)
    root.after(1,pantallazo)
```

Ilustración 15 Sección de código que captura la pantalla.

Fuente: Autor.

La ilustración anterior es una sección del código que se encarga de capturar el lado izquierdo de la pantalla, además de adicionar la fecha y hora de manera gráfica para finalmente guardar dicha captura en un formato de imagen JPG con el nombre de “captura”.

```python
def reconocimiento():
    TOLERANCE=0.6

    frame=face_recognition.load_image_file("captura.jpg")
    frame = cv2.cvtColor(frame,cv2.COLOR_BGR2RGB)


    face_locations = face_recognition.face_locations(frame)
    face_encodings = face_recognition.face_encodings(frame, face_locations)
```

Ilustración 16 Sección de código que procesa la imagen creada por la función “pantallazo”.

(Nombre de función dentro del código)

Fuente: Autor.

Para la ilustración 10, en esta parte del código se procesa la imagen creada en la función “pantallazo”, haciendo una conversión a RGB con el fin de asegurar que el espacio de color es el indicado, esto debido a que la librería face-recognition solo trabaja con escala de grises y RGB.

Además, el método face_locations se encarga de encontrar las caras en la imagen, face_encodings codifica dichas caras en una lista que luego será compara con las listas de caras codificadas al inicio del código, provenientes de la base de datos.

```
for (top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):

    matches = face_recognition.compare_faces(known_face_encodings, face_encoding,TOLERANCE)

    name = "desconocido"

    face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
    best_match_index = np.argmin(face_distances)
    if matches[best_match_index]:
        name = known_face_names[best_match_index]


        cv2.rectangle(frame, (left, top), (right, bottom), (149,168, 11), 2)
        cv2.rectangle(frame, (left, bottom - 20), (right, bottom), (149,168, 11), cv2.FILLED)
        font = cv2.FONT_HERSHEY_DUPLEX
        cv2.putText(frame, name, (left + 6, bottom - 6), font, 0.5, (255, 255, 255), 1)
        cv2.putText(frame,"RECONOCIDO",(400,590),cv2.FONT_HERSHEY_SIMPLEX,1.2,(76, 177, 34),4)
        cv2.imshow("Ventana de reconocimiento", frame)
        reloj = datetime.now()
        cv2.imwrite("registros/VC_"+str(today)+"_"+str(name)+"_"+str(reloj.microsecond)+".jpg",frame)

    else:
        cv2.putText(frame,"DESCONOCIDO",(400,630),cv2.FONT_HERSHEY_SIMPLEX,1.2,(36, 28, 237),4)
        cv2.imshow("Ventana de reconocimiento", frame)
        reloj = datetime.now()
        cv2.imwrite("registros/VC_"+str(today)+"_"+str(name)+"_"+str(reloj.microsecond)+".jpg",frame)
```

Ilustración 17 Sección de código encargada de la identificación facial

Fuente: Autor.

En la sección de código de la ilustración 11 que hace parte de la función “reconocimiento”, se realiza la identificación facial haciendo uso del método face_distance, encargado de hacer el cálculo matemático euclidiano que entrega la distancia entre cada marca facial de referencia (landmarks), para que mediante el método compare_faces se comparen las distancias entre landmarks de la cara a identificar y las caras conocidas de la base de datos. Finalmente, si las distancias entre la cara a identificar y alguna de las caras de la base de datos presenta similitudes se determina que es una cara conocida, de lo contrario el algoritmo asume que se trata de una cara desconocida. Adicional a esto e independientemente de ser una cara reconocida o no, se guarda un registro de la identificación facial de manera gráfica en una carpeta con el nombre de “registros” y

es aquí donde se almacena la información de la persona identificada y/o desconocida junto con la hora y la fecha.

Gran parte del código anterior fue reutilizado para programar algoritmo del servidor, con el fin de detectar si la nueva cara conocida que va a ser agregada a la base de datos ya existe, de manera que se pueda evitar crear dos identidades con un mismo rostro.

El resto del código hace parte del diseño gráfico de la interfaz y no es relevante que sea mostrado en alguna sección del libro, sin embrago, en los anexos de este libro se encuentran los códigos completos programados para cliente y servidor tanto de reconocimiento facial como de interfaces gráficas. Cabe mencionar que el código en su totalidad fue programado de manera funcional.

Para hacer que esta herramienta de reconocimiento facial sea interactiva con el usuario se crea una interfaz muy sencilla y que tan solo con hacer un clic realice la tarea de reconocimiento facial en tiempo real.

En la siguiente imagen se muestra la interfaz implementada.

Ilustración 18 Fig. 6 Interfaz simple para reconocimiento facial en tiempo real.

Fuente: Autor.

Esta interfaz es tan pequeña como la imagen la muestra ya que no necesita ocupar un mayor espacio en la pantalla del equipo de cómputo, consta de un título, una imagen un botón de evento, fecha y hora que se actualiza cada segundo.

Al cliquear el botón “identificar” comienza todo el proceso mencionado anteriormente: cargar base de datos, codificar caras conocidas, capturar pantalla, generar imagen, encontrar caras en la nueva imagen, comparar la cara de la ventana de video llamada con las caras de la base de datos y finalmente determinar si la cara es reconocida o no.

Ahora bien, el algoritmo codifica la nueva cara y la compara con cada una de las caras existentes en la lista de caras codificadas de la base de datos known_faces_encodings = [ ] y al encontrar similitudes determina que es una cara reconocida, por el contrario, al no detectar rasgos faciales similares este determina que es una cara desconocida.

Para entender el tratamiento de la imagen generada de la videollamada, se explica a continuación algunas funciones de la librería face-recognition como lo son face_locations, face_encodings y compare_faces.

A continuación, se muestra una serie de imágenes con el procesamiento de detección, codificación y comparación para verificar y autenticar identidad.

La función face_locations se encarga de encontrar la cara en la imagen y delimitarla en un recuadro con la información de su ubicación superior, inferior, derecha e izquierda y retorna la información en forma de matriz tipo numpy, para esto debemos hacer uso de la librería numpy.

Ilustración 19 Cara localizada mediante la función face_recognition.face_locations.

Fuente: Autor.

Ilustración 20 Matriz numpy que contiene las coordenadas y la información de la cara detectada.

Fuente: Autor.

Ilustración 21 Ejemplo de ilustración para la codificación de la cara detectada con face_recognition.face_encodings.

Fuente: Autor.

La función face_encodings se encarga de codificar la imagen con la cara detectada retornando máximo 128 puntos de rasgos faciales y mínimo 5, cuyo valor máximo toma más tiempo de proceso, mientras el valor mínimo es más rápido, pero menos preciso; estos valores se pueden modificar en los parámetros de la función.

Luego de esto se compara la imagen codificada con cada una de las imágenes codificadas de la base de datos, para esto usamos la función compare_faces.

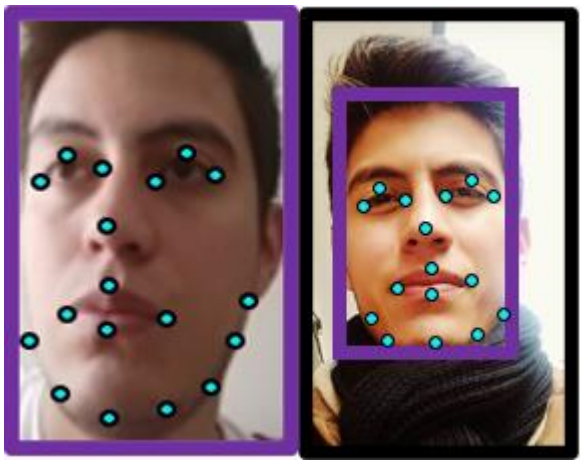


Ilustración 22 Comparativa de rasgos faciales entre la imagen codificada de la captura de pantalla y la imagen existente en la base de datos.

Fuente: Autor.

Cabe mencionar que la imagen de la derecha que pertenece a la base de datos, al igual que la imagen generada por captura de pantalla, realizó todo el proceso mencionado anteriormente.

Finalmente, si los rasgos faciales son similares dibuja un recuadro sobre la cara, un aviso en color verde que dice “RECONOCIDO” y en la parte inferior un nombre que identifica la cara de la siguiente manera.

Ilustración 23 Imagen final con cara reconocida producto de la comparación de rasgos faciales.

Fuente: Autor.

Por el contrario, si los rasgos faciales no corresponden, el resultado final de la imagen es el siguiente.

Ilustración 24 rasgos faciales no similares al momento de realizar la comparación.

Fuente: Autor.

El algoritmo muestra una imagen con un aviso en color rojo con la palabra "DESCONOCIDO" y no encierra la cara pues no identifica a quien pertenece esta cara.

Finalmente, cualquiera de estas imágenes se guarda en una carpeta llamada con el nombre de "registros" y es en esta carpeta donde queda la evidencia de que personas fueron reconocidas y quiénes no.

Para almacenar una cara en la base de datos, se ha dispuesto de una interfaz gráfica que permite a través de una cámara web tomar una fotografía de la cara de la persona y asignarle un nombre de usuario con el cual será identificado cuando el programa lo reconozca. En la siguiente imagen se muestra el diseño de la interfaz y de las funciones que tiene.

Ilustración 25 Interfaz gráfica para el almacenamiento de nuevas caras en la base de datos.

Fuente: Autor.

Esta interfaz consta de un recuadro de video en tiempo real para la cámara web (CAMARA 1), un recuadro de pre visualización de la imagen que será almacenada en la base de datos (FOTO), cuatro botones de eventos.

A continuación, se muestra un ejemplo de la alerta gráfica que se emite luego de presionar los botones TOMAR FOTO o VERIFICAR EN LA BASE DEDATOS.

Ilustración 26 Alerta gráfica para usuario o cara existente dentro de la base de datos de caras reconocidas por el algoritmo de reconocimiento facial.

Fuente: Autor.

Ilustración 27 Alerta gráfica para cara o usuario inexistente dentro de la base de datos de caras reconocidas por el algoritmo de reconocimiento facial.

Fuente: Autor.

Finalmente, la interfaz gráfica cuenta con un pequeño aviso “Esperando foto…” que puede cambiar a “CARA ENCONTRADA” o “CARA NO ENCONTRADA”, esta pequeña alarma permite saber si en la imagen tomada por la cámara web fue encontrada una cara o no para la creación de un usuario. En la parte inferior cuenta con un pequeño manual gráfico que trabaja como guía de trabajo.

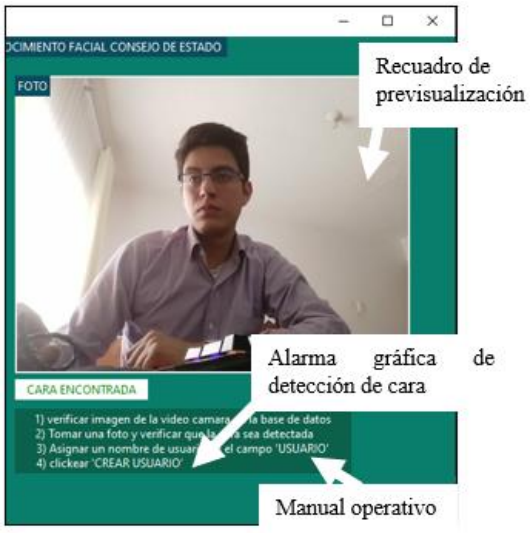


Ilustración 28 Interfaz gráfica luego de tomar una foto para la creación de un usuario.

Fuente: Autor.

Este sistema informático será implementado dentro de una red local con arquitectura cliente servidor, donde en el servidor se encontrará la base de datos con todas las caras almacenadas que el algoritmo reconoce, mientras que en el cliente se encontrará únicamente la interfaz para reconocimiento facial.

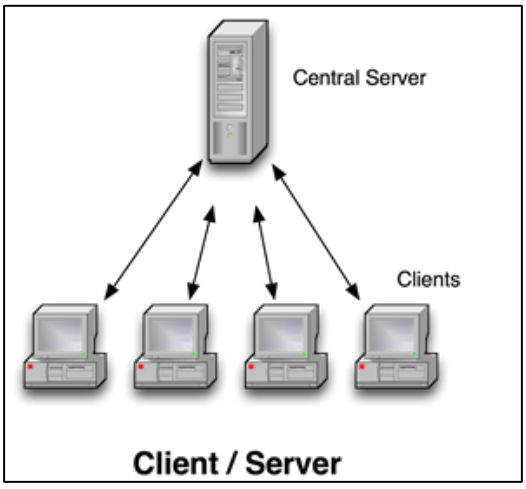


Ilustración 29 red local arquitectura cliente servidor.

Fuente: funcionamiento-redes-lan, NetCloud 2019

La base de datos puede ser accedida mediante carpetas compartidas y es el administrador el encargado de otorgar permisos de accesibilidad a la carpeta que contiene la base de datos de caras reconocidas.

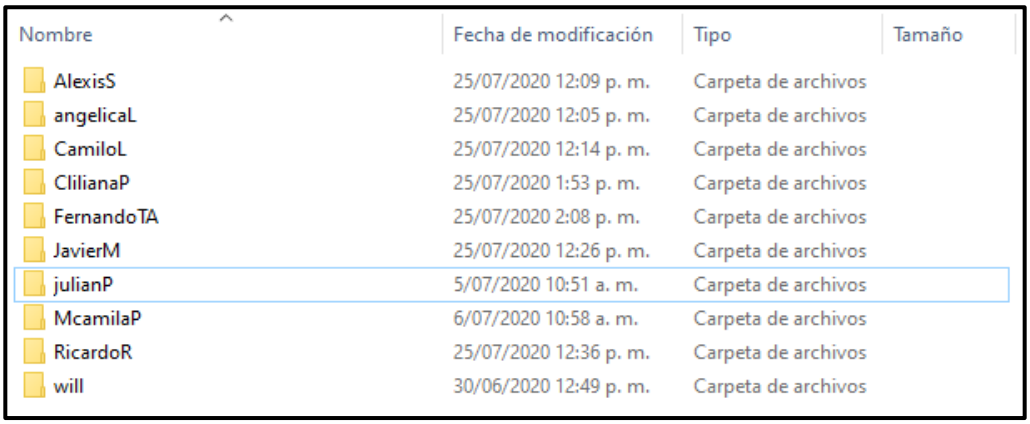
9.3 ETAPA 3

9.3.1 Resultados

En este capítulo se muestran las pruebas realizadas para determinar en qué casos el sistema falla o no está en capacidad de identificar biométricamente una persona, finalmente realizar una medición implementando una matriz de confusión que generalmente se utiliza para sistemas que predicen, es decir sistemas que implementan inteligencia artificial.

Para realizar las pruebas de reconocimiento facial (modo cliente), se seleccionaron diez personas que voluntariamente quisieron participar. Durante cinco días se tomará una imagen de la cara de cada uno de los participantes y se someterá a reconocimiento facial, modificando algunos aspectos como ángulo de posición, iluminación, lentes, maquillaje (mujeres), uso de tapabocas e imágenes en baja resolución, este último aspecto para cuando la velocidad del internet disminuya y la calidad del video durante la video llamada sea de baja resolución.

En primer lugar, para crear la base de datos de caras, cada uno de los participantes debió aportar una foto con una antigüedad mínima de dos años o máxima de cinco años, además asignarles un nombre de usuario con el que serán identificados durante la prueba. Cabe recordar que estas imágenes estarán en el equipo de cómputo configurado como servidor donde se encuentra la base de datos. Una muestra de esto se puede evidenciar en la ilustración 22.



Nombre	Fecha de modificación	Tipo	Tamaño
AlexisS	25/07/2020 12:09 p. m.	Carpeta de archivos	
angelicaL	25/07/2020 12:05 p. m.	Carpeta de archivos	
CamiloL	25/07/2020 12:14 p. m.	Carpeta de archivos	
ClilianaP	25/07/2020 1:53 p. m.	Carpeta de archivos	
FernandoTA	25/07/2020 2:08 p. m.	Carpeta de archivos	
JavierM	25/07/2020 12:26 p. m.	Carpeta de archivos	
julianP	5/07/2020 10:51 a. m.	Carpeta de archivos	
McamilaP	6/07/2020 10:58 a. m.	Carpeta de archivos	
RicardoR	25/07/2020 12:36 p. m.	Carpeta de archivos	
will	30/06/2020 12:49 p. m.	Carpeta de archivos	

Ilustración 30 Base de datos con los directorios que contienen las caras conocidas de cada uno de los usuarios en un formato de imagen jpg.

Fuente: Autor.

Para cada una de las pruebas se genera un collage con la imagen diaria y sus modificaciones para ser sometido a reconocimiento facial, esto con el fin de no hacer tan extenso este documento.

A continuación, se muestra un collage ejemplo de las caras dentro de la base de datos y sus respectivos nombres de usuario.

Ilustración 31 Fotos que conforman la base de datos de caras conocidas aportadas por los participantes en formato jpg.

Fuente: Autor.

Primera prueba, caras de frente a la cámara y sin ningún tipo de modificación.

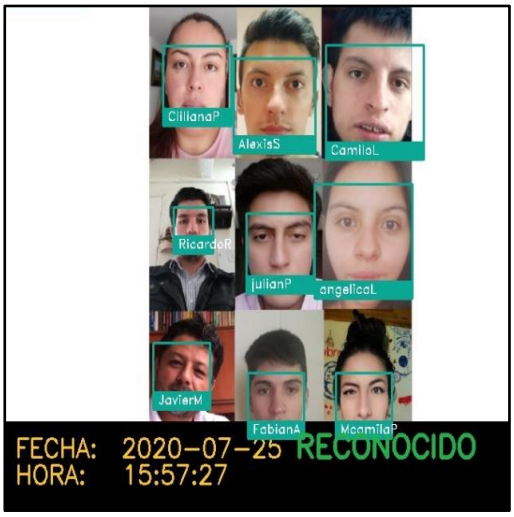


Ilustración 32 primera prueba. Rostros reconocidos en su totalidad.

Fuente: Autor.

Segunda prueba, se pide a los participantes que usen maquillaje y gafas con lentes tanto transparentes como oscuros.

Ilustración 33 Segunda prueba. Se evidencia un error de identificación y el sistema no puede reconocer uno de los participantes que posee lentes oscuros.

Fuente: Autor.

En la segunda prueba se observa que la cara dentro del recuadro amarillo la identificación es incorrecta, ya que el sistema reconoce de manera errónea al participante RicardoR identificándolo como FabianA, esto se debe a que en la base de datos la imagen de la cara del participante RicardoR es de baja resolución, se adicionan gafas y durante la video llamada se modifica la iluminación. A continuación se muestra la comparativa de la imagen de base de datos y la prueba uno y dos.

Ilustración 34 Comparativa de imágenes. Base de datos (baja resolución) // prueba uno (identificación correcta) // prueba dos (identificación incorrecta)

Fuente: Autor.

Ahora para el participante identificado como JavierM, en la segunda prueba el sistema no logra reconocerlo, esto debido a que los lentes oscuros eliminan gran parte de sus rasgos faciales y la iluminación en mitad de su rostro es opaca. Esto hace que el sistema no logre encontrar una similitud en los rasgos faciales de la cara que se encuentra en la base de datos.

A continuación se realiza la comparativa de imágenes de la cara de base de datos y prueba dos para el participante JavierM.

Ilustración 35 comparativa de imágenes. Base de datos // prueba dos con lentes (identificado) // prueba dos con lentes oscuros (desconocido).

Fuente: Autor.

A partir de esta prueba se puede determinar que se pueden presentar errores como los mencionados anteriormente y que posiblemente en las pruebas faltantes sucederán de igual forma.

Para la tercera prueba se evidencian algunos fallos similares a la prueba anterior, donde se puede ver que al modificar u obstruir rasgos faciales el sistema confunde identidades o sencillamente no puede detectar la forma de un rostro.

Ilustración 36 Tercera prueba. El sistema confunde identidades o no detecta rostros cuando los rasgos faciales son obstruidos.

Fuente: Autor.

Para los recuadros amarillos se evidencia una confusión de identidades donde a la participante McamilaP la reconoce como ClilianaP y al participante CamiloL lo identifica como FabianA, esto se debe a que lo lentes y tapabocas eliminan rasgos faciales que conllevan a que el sistema se equivoque o confunda identidades. Por otra parte, en los recuadros morados el sistema no logra percibir la presencia de un rostro durante la videollamada debido a que los tapabocas cubren rasgos faciales importantes para la detección de una cara.

Para la cuarta prueba se adicionan dos participantes más a la base de datos y se elimina al participante julianP, pero continúa dentro del proceso de pruebas.

Ilustración 37 participantes adicionadas a la base de datos.

Fuente: Autor.

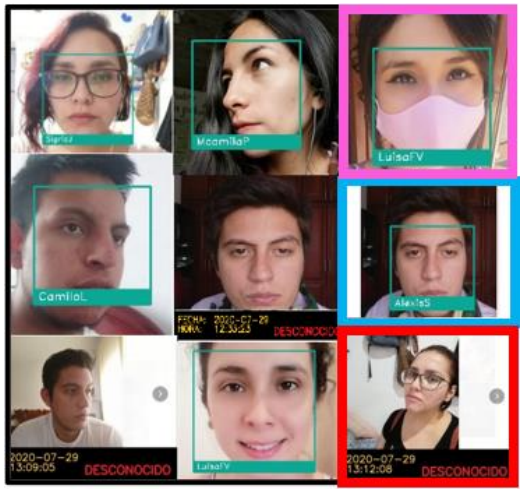


Ilustración 38 Prueba número cuatro. Se presentan Algunos errores de identificación.

Fuente: Autor.

Esta prueba arroja nuevos resultados donde se evidencia que el tapabocas color rosa no limita al sistema para identificar correctamente una cara, como lo es el caso del recuadro color fucsia. En el recuadro azul el sistema identifica erróneamente pues debería mostrar que es una cara "DESCONOCIDA", pero lo muestra como AlexisS, se hace un ajuste a la sensibilidad de reconocimiento y se logra corregir este error. En el recuadro rojo esta cara pertenece a la

participante SigridJ sin embargo el sistema no logra identificarla correctamente y la detecta como una persona desconocida, posiblemente sea por factores de iluminación y ángulo de posición que alteran algunos bordes o rasgos faciales.

Para la quinta prueba se pretende disminuir la calidad de la videollamada para determinar si el sistema está en la capacidad de reconocer un rostro cuando la velocidad del internet es lenta, es decir trabajar con una resolución baja.

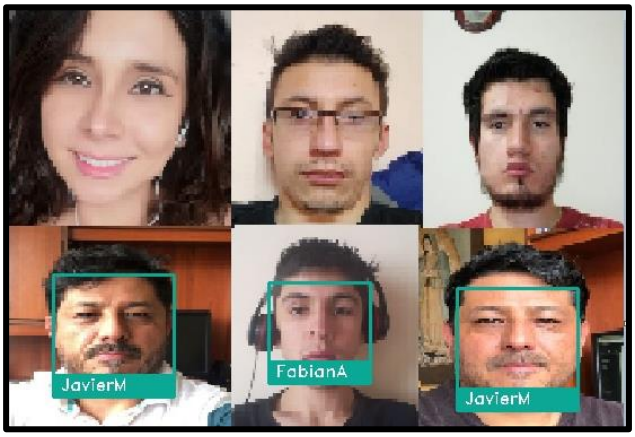


Ilustración 39 Quinta prueba. Reconocimiento facial en videollamada con baja resolución.

Fuente: Autor.

Esta prueba solo reconoce tres de los seis rostros con baja resolución, por lo que se concluye que una buena resolución de videollamada es fundamental para el reconocimiento facial. Finalmente se cuantifica el resultado de las pruebas realizadas en una matriz de confusión.

Una matriz de confusión se emplea en el campo de la inteligencia artificial y aprendizaje automático que en términos generales permite ver cuantos aciertos y errores tiene el modelo al momento del aprendizaje. (Arce, 2019)

La matriz se organiza de la siguiente manera:

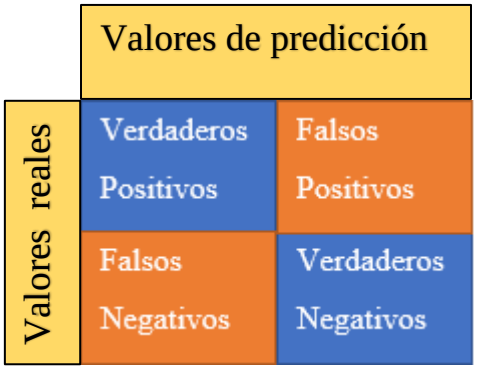


Ilustración 40 Organización matriz de confusión

Fuente: health Big Data, Arce 2019

Donde:

Verdadero positivo: El valor real es positivo y la predicción es positiva.

Verdadero negativo: El valor real es negativo y la predicción es negativa.

Falso negativo: El valor real es positivo y la predicción es negativa.

Falso positivo: El valor real es negativo y la predicción es positiva.

Ahora bien, para la métrica se utiliza una serie de valores como lo son precisión, recall, F1 score y accuracy definidos de la siguiente manera:

Precisión: porcentaje de casos positivos detectados.

Se calcula como: verdadero positivo / (verdadero positivos+ Falsos positivos)

Recall: valor que indica la capacidad del modelo para distinguir los casos positivos de los negativos.

Se calcula como: Verdadero Positivo / (Verdadero Positivo + Falsos Negativos).

F1- score: Este es un valor que resume la precisión y recall en una sola métrica.

Se calcula como: 2 * (recall*precisión) / (recall + precisión).

Accuracy: Es un valor que mide el porcentaje de casos que el modelo de clasificación ha acertado.

Se calcula como: (Verdaderos Positivos + Verdaderos Negativos) / (VP + VN + FP +FN)

Cabe recordar que fueron cuarenta y dos imágenes de videollamadas sometidas a reconocimiento facial donde los resultados quedaron de la siguiente manera.

		PREDICCIÓN	
		POSITIVO	NEGATIVO
ACTUAL	POSITIVO	29	2
	NEGATIVO	4	2

Ilustración 41 Matriz de confusión para el modelo de reconocimiento facial.

Fuente: Autor.

$$precision = \frac{29}{29+4} = 0.87$$

$$recall = \frac{29}{29+2} = 0.93$$

$$F1 = 2\ \frac{precision\ .\ recall}{precision + recall} = \frac{(0.87)(0.93)}{0.87+0.93} = 0.88$$

0.88

$$accuracy = \frac{2+29}{29+4+2+2} = 0.83$$

Para un total de treinta y siete imágenes sometidas a identificación facial, sin embargo cinco imágenes no fueron incluidas dentro de la métrica ya que la red neuronal no pudo detectar rostros porque gran parte de los rasgos faciales fueron obstruidos por tapabocas, gafas oscuras o baja resolución.

Estas pruebas se realizaron con el fin de detectar en qué casos el sistema de reconocimiento no está en capacidad de identificar facialmente a una persona, lo cual abre nuevas perspectivas de recomendaciones a trabajos futuros con la línea de detección biométrica facial, donde se sugiere crear y entrenar una red neuronal que identifique una persona teniendo en cuenta los posibles accesorios faciales como pueden ser tapabocas, gafas oscuras, maquillaje, entre otros.

10 CONCLUSIONES

-Se logra ensamblar un software de reconocimiento facial programado en lenguaje Python, que funciona para el sistema operativo Windows 10 (sistema del que hace uso la corte consejo de estado), donde se emplean herramientas de inteligencia artificial como visión por computador (librería Opencv) y redes neuronales (librería Dlib), haciendo de este, un software de alta fiabilidad, fácil ejecución y potencialmente implementable, que además de ser una herramienta informática, también contribuye con la detección de fraudes en la suplantación de identidades y permite autenticar identidades durante las audiencias en los procesos legales.

-El sistema informático de reconocimiento facial está en la capacidad de identificar a una persona en tiempo real, pero presenta algunas excepciones como las pruebas realizadas lo demuestran, por lo cual se recomienda al usuario ejecutar varias veces el algoritmo durante la videollamada en la audiencia legal, con el fin de corroborar que realmente el reconocimiento facial por parte del algoritmo corresponde y concuerda con las identidades de la base de datos y las identidades manifestadas en la audiencia.

-La resolución es fundamental para que el sistema de reconocimiento facial pueda identificar de manera acertada el rostro de una persona. Al ser baja la resolución en una imagen digital se pierde información primordial necesaria como lo son los rasgos faciales, que en el campo de la inteligencia artificial y la visión por computador se traduce en bordes compuestos por pixeles y son estos pixeles los que poseen la información que posteriormente la red neuronal debe clasificar para detectar una cara y basada en su entrenamiento decidir si la reconoce o no. Por esta razón se debe contar con una alta calidad de resolución, tanto en la imagen de la videollamada como en las imágenes de caras conocidas que se almacenan en la base de datos.

-El sistema de reconocimiento facial presenta algunos desaciertos o confusiones al momento de identificar una cara debido a las distorsiones en rasgos faciales expuestos en las pruebas, esto hace que el cálculo matemático hecho por algoritmo euclidiano, genere identificaciones erróneas al encontrar mediciones similares entre las marcas faciales de referencia (landmarks), motivo por el cual la cara de la persona que interviene en la audiencia a través de la videollamada, debe estar despejada y con buena iluminación de manera que permita al sistema tener una mayor precisión en la identificación.

-El funcionamiento cliente servidor opera de manera correcta sin ningún problema y con una excelente velocidad de conexión con la base de datos para una red local LAN, siempre y cuando la conexión entre equipos sea mediante el puerto ethernet que ofrece rapidez y poca perdida en los paquetes de datos.

11 RECOMENDACIONES

1. Es muy importante contar con la autorización o permisos de tratamiento de datos de las personas que sean sometidas a reconocimiento facial por este software, ya que el manejo de datos tanto personales como biométricos está regulado por ley, por lo cual se recomienda implementar un formulario para solicitar permisos, términos y condiciones para el manejo de datos y de esta manera no incurrir en una falta grave relacionada con la violación a la privacidad de datos personales.
2. Las caras almacenadas en la base de datos no deben estar obstruidas por ningún accesorio, en lo posible tener una imagen de la cara de frente con buena iluminación y una resolución aceptable.
3. La conexión de la red local LAN debe ser a través de los puertos de conexión ethernet de los equipos de cómputo para poder contar con una alta velocidad de conectividad entre el cliente y la base de datos del servidor.

12 REFERENCIAS

Presente y futuro del reconocimiento facial. (n.d.). Retrieved June 16, 2020, from https://www.padigital.es/tendencias-digitales/historia-y-evolucion-del-reconocimiento-facial.html

Dlib C++ Library: High Quality Face Recognition with Deep Metric Learning. (n.d.). Retrieved June 16, 2020, from http://blog.dlib.net/2017/02/high-quality-face-recognition-with-deep.html

Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning. (n.d.). Retrieved June 16, 2020, from https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78

Ketkar, N. (2017). Deep Learning with Python. In *Deep Learning with Python*. https://doi.org/10.1007/978-1-4842-2766-4

Lutz, M. (2007). Learning Python. In *Icarus*. https://doi.org/10.1016/0019-1035(89)90077-8

Bradski, G., & Kaehler, A. (2008). Learning OpenCV. In *Learning OpenCV*. https://doi.org/10.1109/MRA.2009.933612

Zelinsky, A. (2009). Learning OpenCV—Computer Vision with the OpenCV Library. In *IEEE Robotics and Automation Magazine*. https://doi.org/10.1109/MRA.2009.933612

Programming python. (1997). *Computers & Mathematics with Applications*. https://doi.org/10.1016/s0898-1221(97)82952-5

Howse, J. (2013). OpenCV Computer Vision with Python. In *Cs_Python_in*. https://doi.org/10.1017/CBO9781107415324.004

Itseez. (2012). *OpenCV - Open Source Computer Vision*. Online Webpage.

Xie, Z., Li, J., & Shi, H. (2019). A Face Recognition Method Based on CNN. *Journal of Physics: Conference Series*. https://doi.org/10.1088/1742-6596/1395/1/012006

Yaswanth, V. V. S., & Devi, T. (2019). Automatic emotion recognition using facial expression by python. *Test Engineering and Management*.

ADIEGO RODRIGUEZ, J. (2007). *books google*. Obtenido de https://books.google.es/books?id=FfEfCB-hXCgC&pg=PT297&dq=base+datos+relacional+codd&hl=es&sa=X&ved=0ahUKEwjixfLYxcPXAhXMWhQKHQImAfUQ6AEIQjAF#v=onepage&q=base%20datos%20relacional%20codd&f=false

Amos, B. (septiembre de 2016). *ResearchGate*. Obtenido de ResearchGate: https://www.researchgate.net/figure/The-68-landmarks-detected-by-dlib-library-This-image-was-created-by-Brandon-Amos-of-CMU_fig2_329392737

Arce, J. I. (26 de julio de 2019). *health Big Data*. Obtenido de https://www.juanbarrios.com/matriz-de-confusion-y-sus-metricas/

Boyko, N., Basystiuk, O., & Shakhovska, N. (2018). Performance Evaluation and Comparison of Software for Face Recognition, Based on Dlib and Opencv Library. *IEEE explore*.

contabal. (2016). Obtenido de https://www.contaval.es/que-es-la-vision-artificial-y-para-que-sirve/

Duque, R. G. (2020). Python para todos. En R. G. Duque, *Python para todos.*

Geitgey, A. (2017). *Readthedocs*. Obtenido de Readthedocs: https://face-recognition.readthedocs.io/en/latest/face_recognition.html

Guay, M. (2010). *how to geek*. Obtenido de https://www.howtogeek.com/howto/17528/change-the-user-interface-language-in-ubuntu/

Inovation, A. (2019). *Atria Inovation*. Obtenido de https://www.atriainnovation.com/que-son-las-redes-neuronales-y-sus-funciones/

King, D. (2017). *http://blog.dlib.net/*. Obtenido de http://blog.dlib.net/: http://blog.dlib.net/

Na8. (28 de noviembre de 2019). *como funcionan las convolutional neural networks?* Obtenido de aprende machine learnign: https://www.aprendemachinelearning.com/como-funcionan-las-convolutional-neural-networks-vision-por-ordenador/

NetCloud. (2019). Obtenido de https://netcloudengineering.com/funcionamiento-redes-lan/

Opencv.org. (2020). *Opencv*. Obtenido de Opencv: https://opencv.org/about/

PADigital. (27 de noviembre de 2018). *padigital.es*. Obtenido de https://www.padigital.es/tendencias-digitales/historia-y-evolucion-del-reconocimiento-facial.html#:~:text=Los%20inicios%20del%20reconocimiento%20facial,trav%C3%A9s%20de%20la%20tabla%20RAND.

Quintana, J. (mayo de 2018). *medium.com*. Obtenido de medium.com: https://medium.com/datos-y-ciencia/modelos-cnn-en-la-clasificaci%C3%B3n-de-im%C3%A1genes-cl%C3%A1sicas-y-modernas-d072a6718689

Rajesh, K. M., & Naveenkumar, M. (2016). A robust method for face recognition and face emotion detection system using support vector machines. *IEEE explore*.

Rivera, M. (Agosto de 2019). *resnet.md*. Obtenido de resnet.md: http://personal.cimat.mx:8181/~mrivera/cursos/aprendizaje_profundo/resnet/resnet.html

Russell, S. J., & Norvig, P. N. (2009). Artificial intelligence: a modern approach. En S. J. Russell, & P. N. Norvig.

Sampieri, R. (2014). *Metodología de la investigación 6ta Edición.* McGRAW-HILL / INTERAMERICANA EDITORES, S.A. DE C.V.

significados.com. (2020). Obtenido de https://www.significados.com/software/

Sphinx. (2019). *https://pyautogui.readthedocs.io/en/latest/*. Obtenido de https://pyautogui.readthedocs.io/en/latest/: https://pyautogui.readthedocs.io/en/latest/

tecnologia informatica. (2020). Obtenido de https://www.tecnologia-informatica.com/algoritmo-definicion/

Thilaga, P. J., Khan, B. A., Jones, A., & Kumar, N. K. (2018). Modern Face Recognition with Deep Learning. *IEEE explore*.

zator.com. (2020). Obtenido de zator.

13 ANEXOS