



**Sistema de control de acceso mediante identificación y verificación facial
fundamentado en algoritmos de aprendizaje automático y redes neuronales**

Ibarra Flores, William Iván

Departamento de Eléctrica, Electrónica y Telecomunicaciones

Carrera de Ingeniería en Electrónica y Telecomunicaciones

Trabajo de titulación previo a la obtención del título de Ingeniero en Electrónica y
Telecomunicaciones







Dr. Olmedo Cifuentes, Gonzalo Fernando

28 de agosto del 2020

Document Information

Analyzed document	Proyecto_de_grado_William_Ibarra.pdf (D78418907)
Submitted	8/31/2020 8:37:00 PM
Submitted by	Olmedo Cifuentes Gonzalo Fernando
Submitter email	gfolmedo@espe.edu.ec
Similarity	1%
Analysis address	gfolmedo.espe@analysis.orkund.com

Sources included in the report

SA	Universidad de las Fuerzas Armadas ESPE / Tesis Israel Obando.pdf Document Tesis Israel Obando.pdf (D50917962) Submitted by: vgproanio@espe.edu.ec Receiver: vgproanio.espe@analysis.orkund.com		5
SA	Proyecto_TFG-1-53.pdf Document Proyecto_TFG-1-53.pdf (D53240258)		1
SA	B2.592_20192_PEC 2: Evoluci◊n trabajo final y memoria_12556911.txt Document B2.592_20192_PEC 2: Evoluci◊n trabajo final y memoria_12556911.txt (D73268443)		2
SA	1569252904_tesis_shendry_con cambios sugeridos.pdf Document 1569252904_tesis_shendry_con cambios sugeridos.pdf (D55939687)		3
W	URL: https://rdu.unc.edu.ar/bitstream/handle/11086/11694/Pastore.pdf?sequence=1&isAllowed=y Fetched: 4/25/2020 10:32:03 AM		2
SA	M2.880_20192_PEC2 - Estado del arte o an◊lisis de mercado del proyecto_11931150.txt Document M2.880_20192_PEC2 - Estado del arte o an◊lisis de mercado del proyecto_11931150.txt (D66130249)		1



Firmado electr3nicamente por:
GONZALO FERNANDO
OLMEDO CIFUENTES

Dr. Olmedo Cifuentes, Gonzalo Fernando
Profesor Titular Principal
C. C.1711696342



DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que el trabajo de titulación, “**Sistema de control de acceso mediante identificación y verificación facial fundamentado en algoritmos de aprendizaje automático y redes neuronales**” fue realizado por el señor **Ibarra Flores, William Iván** el cual ha sido revisado y analizado en su totalidad por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 28 de agosto del 2020



Firmado electrónicamente por:
**GONZALO FERNANDO
OLMEDO CIFUENTES**

.....
Dr. Olmedo Cifuentes, Gonzalo Fernando

C. C.1711696342



DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

RESPONSABILIDAD DE AUTORÍA

Yo, **Ibarra Flores, William Iván**, con cédula de ciudadanía n°0604423020, declaro que el contenido, ideas y criterios del trabajo de titulación: **Sistema de control de acceso mediante identificación y verificación facial fundamentado en algoritmos de aprendizaje automático y redes neuronales** es de mi autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Sangolquí, 28 de agosto del 2020

Ibarra Flores, William Iván

C.C.: 0604423020



DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN DE PUBLICACIÓN

Yo **Ibarra Flores, William Iván**, con cédula de ciudadanía n°0604423020, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **Sistema de control de acceso mediante identificación y verificación facial fundamentado en algoritmos de aprendizaje automático y redes neuronales** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, 28 de agosto del 2020

Ibarra Flores, William Iván

C.C.: 0604423020

DEDICATORIA

A mi mami querida Cecy, que con su amor, comprensión y aliento me ayuda a seguir adelante en esta travesía denominada vida.

A mi padre, por apoyarme en el transcurso de mi vida universitaria y en los momentos que más lo necesito.

A mis hermanos Deyvid y Verito por estar conmigo en los tiempos buenos y malos, están en mi corazón, los quiero mucho.

Con mucho cariño para las personas que amo y respeto

William Iván Ibarra Flores

AGRADECIMIENTO

Muy orgulloso de poder hacer mención de todos a quienes agradezco el poder llegar al cumplimiento de esta meta tan anhelada que es mi título profesional a través de este trabajo de tesis, en honor a tal orgullo, deseo expresar mi agradecimiento con los siguientes:

Agradezco mi Dios por darme la oportunidad de vivir, permitiendo que la comparto con personas maravillosas y manteniéndome firme para que culmine mi carrera universitaria.

Gracias a mis padres Arnulfo y Cecy por ser pilares fundamentales en mi vida y educación, por creer y confiar en mí, siendo mis mentores y testigos de la persona que están formando.

Gracias a mis hermanos Deyvid y Verito porque sé, que cuento con ellos en todo momento y hacen de mi vida más amena con sus ocurrencias y consejos.

Agradezco a la Universidad de la Fuerzas Armadas por darme la oportunidad de cumplir mi sueño de ser Ingeniero, a los docentes de tan alta calidad, en especial a mis tutores de tesis Dr. Gonzalo Olmedo e Ing. Nancy Paredes, mi dios les pague por permitirme realizar este trabajo con ustedes y brindarme sus conocimientos.

Finalmente, agradezco a mis amigos por todas las experiencias vividas dentro de esta etapa estudiantil.

Muchas gracias a todos

William Iván Ibarra Flores

Índice de contenidos

Resumen	15
Capítulo 1	17
Introducción	17
Antecedentes	17
Justificación e Importancia	19
Alcance del proyecto.....	21
Objetivos.....	21
General	21
Específicos.....	21
Resumen de contenidos	22
Capítulo 2	23
Marco teórico	23
Introducción a la biometría.....	23
Sistemas biométricos.....	25
La Inteligencia artificial (IA)	27
Aprendizaje profundo	27
Visión por computador.....	28
Reconocimiento facial.....	28
Aprendizaje automático (ML)	29
K- vecinos más cercanos (KNN).....	31
Máquina de vectores de soporte (SVM).....	32
Redes neuronales.....	34
Red neuronal artificial.....	34

Red neuronal convolucional (ConvNet)	38
Software y hardware	41
OpenCV	41
Python	41
Face recognition	42
Jetson nano	42
Cámara CSI	43
Capítulo 3	44
Diseño e implementación	44
Descripción	44
Módulo de detección de tono	45
Módulo de registro	48
Adquisición de datos	48
Procedimiento para el registro de rostros	52
Módulo de reconocimiento facial	54
Extracción de características	55
Preprocesamiento de imagen	57
Detección de rostro	59
Creación de incrustaciones	64
Entrenamiento	70
Entrenamiento con KNN	72
Entrenamiento con SVM	73
Predicción	74
Predicción con KNN	76
Predicción con SVM	78
Módulo de acceso	80
Incorporación sistema – chapa	80
Historial de ingreso	83
Integración	87

	10
Descripción general del proceso.....	87
Capítulo 4	90
Pruebas y resultados	90
Pruebas	90
Parámetros de evaluación	92
Contexto	92
Desarrollo de pruebas.....	94
Pruebas con KNN.....	94
Prueba 1 – distancia	94
Prueba 2 – iluminación.....	98
Pruebas con SVM	101
Prueba 1 – distancia	101
Prueba 2 – iluminación.....	104
Análisis comparativo	107
Prueba distancia.....	107
Prueba de iluminación	109
Capítulo 5	112
Conclusiones y recomendaciones.....	112
Conclusiones	112
Recomendaciones	113
Trabajos futuros	114
Referencias bibliográficas	115
Anexos.....	119

Índice de tablas

Tabla 1	Características biológicas para la identificación	25
Tabla 2	Librerías para la detección de tono	46
Tabla 3	Características de los materiales empleados en el registro de usuarios	48
Tabla 4	Características de los materiales empleados en el reconocimiento facial	55
Tabla 5	Librerías empleadas en el módulo de extracción de características.....	57
Tabla 6	Parámetros del modelo empleado	63
Tabla 7	Distribución de los 68 puntos de referencia en un rostro.....	65
Tabla 8	Modelos pre-entrenados	69
Tabla 9	Librerías de entrenamiento	71
Tabla 10	Librerías de predicción.....	75
Tabla 11	Software de instalación para el historial de ingreso.....	83
Tabla 12	Librerías empleadas en la inscripción de usuarios	84
Tabla 13	Argumentos de conexión	85
Tabla 14	Campos de la tabla personal creada en MySQL	86
Tabla 15	Pruebas de distancia	91
Tabla 16	Pruebas de intensidad de luz.....	91
Tabla 17	Métricas de evaluación	92
Tabla 18	Sujetos de prueba.....	93
Tabla 19	Resultado de precisión con KNN a diferentes distancias	96
Tabla 20	Resultado de precisión con KNN a diferente intensidad de Luz	99
Tabla 21	Resultado de precisión con SVM a diferentes distancias	102
Tabla 22	Resultado de precisión con SVM a diferentes direcciones de iluminación ..	105
Tabla 23	Resultados de precisión con KNN - SVM.....	107
Tabla 24	Resultados de tiempo de clasificación KNN - SVM	108

Tabla 25 Resultados de precisión con KNN - SVM.....	109
Tabla 26 Resultados de tiempo de clasificación KNN - SVM	111

Índice de figuras

Figura 1 Tres formas básicas en que una persona puede demostrar su identidad.....	24
Figura 2 Esquema general de todo sistema biométrico	26
Figura 3 Modelo básico de machine learning	30
Figura 4 Método de clasificación KNN.....	31
Figura 5 Hiperplano de Separación Óptimo implementado para dos clases	33
Figura 6 Mapeo de datos a un espacio de mayor dimensión.....	34
Figura 7 Modelo de una neurona artificial.....	35
Figura 8 Red neuronal feedback	37
Figura 9 Red neuronal feedforward.....	37
Figura 10 Contraste de red neuronal normal vs red neuronal convolucional	38
Figura 11 Conexión entre capas ConvNet.....	39
Figura 12 Arquitectura de una ConvNet en la detección de un objeto	40
Figura 13 Arquitectura de OpenCV	41
Figura 14 Módulo Jetson nano	42
Figura 15 Cámara raspberry pi V2	43
Figura 16 Diagrama de bloques del sistema de control de acceso.....	45
Figura 17 Diagrama de componentes del programa bell.py.....	46
Figura 18 Diagrama de flujo del programa bell.py	47
Figura 19 Diagrama de componentes del programa build_face_dataset.py	49
Figura 20 Detección de rostro mediante OpenCV y modelo de Haar	50
Figura 21 Diagrama de flujo del programa built_face_dataset.py	51

Figura 22	Ruta donde se almacenarán las imágenes (base de datos)	52
Figura 23	Ejecución del programa build_face_dataset.py	52
Figura 24	Base de datos creada con un integrante de la BD.....	53
Figura 25	Programas del módulo de reconocimiento facial	54
Figura 26	Etapas del submódulo de extracción de características	56
Figura 27	Diagrama de componentes del programa face_extract.py.....	56
Figura 28	Imagen luego de las capas de reducción	60
Figura 29	Localización de las regiones de interés (ROI)	60
Figura 30	Detección de rostro mediante CNN	61
Figura 31	Estructura de la CNN empleada	62
Figura 32	68 puntos de referencia en un rostro	65
Figura 33	Mapeo de 68 puntos faciales en una imagen de la base de datos.....	66
Figura 34	Aplicación del descriptor 128D a una imagen de la BD	67
Figura 35	Proceso de entrenamiento de la Resnet-29.....	68
Figura 36	Diagrama de componentes del programa face_train.py.....	71
Figura 37	Diagrama de proceso de predicción	75
Figura 38	Predicción de rostro con el método KNN.....	77
Figura 39	Predicción con el método SVM.....	79
Figura 40	Predicción de una persona con el método SVM	79
Figura 41	Pines de conexión de la chapa magnética.....	80
Figura 42	Diagrama de conexión chapa-Jetson Nano	81
Figura 43	Diagrama de los componentes del programa enable.py	82
Figura 44	Diagrama de los componentes del programa inscription.py.....	84
Figura 45	Inscripción del usuario William Ibarra en MySql.....	86
Figura 46	Diagrama de componentes del sistema de control de acceso	87
Figura 47	Sistema de control de acceso.....	88

Figura 48	Vista frontal del sistema de control de acceso.....	89
Figura 49	Vista superior del sistema de control de acceso.....	89
Figura 50	Entorno de pruebas.....	93
Figura 51	Sujeto de prueba identificado a Distancia 1.....	94
Figura 52	Puntaje de distancia facial vs Puntaje porcentual.....	95
Figura 53	Porcentaje de coincidencia a Distancia 1.....	95
Figura 54	Precisión (%) en la prueba de Distancias.....	97
Figura 55	Tiempos de clasificación (seg) en la prueba de Distancias.....	97
Figura 56	Sujeto de prueba identificado a iluminación 1.....	98
Figura 57	Porcentaje de coincidencia a iluminación 1.....	98
Figura 58	Precisión (%) en la prueba de Luz.....	100
Figura 59	Tiempos de clasificación (seg) en la prueba de Iluminación.....	100
Figura 60	Sujeto de prueba identificado a Distancia 3.....	101
Figura 61	Porcentaje de coincidencia a Distancia 3.....	102
Figura 62	Precisión (%) en la prueba de Distancias.....	103
Figura 63	Tiempos de clasificación (seg) en la prueba de Distancias.....	103
Figura 64	Sujeto de prueba identificado a iluminación 2.....	104
Figura 65	Porcentaje de coincidencia a iluminación 2.....	104
Figura 66	Precisión (%) en la prueba de Iluminación.....	106
Figura 67	Tiempos de clasificación (seg) en la prueba de Luz.....	106
Figura 68	Rendimiento de cada método durante la prueba de distancias.....	107
Figura 69	Tiempo de clasificación durante la prueba de distancias.....	109
Figura 70	Rendimiento de cada método durante la prueba de distancias.....	110
Figura 71	Tiempo de clasificación durante la prueba de distancias.....	111

Resumen

Este proyecto presenta el diseño e implementación de un sistema de control de acceso biométrico no intrusivo basado en el reconocimiento facial mediante algoritmos de aprendizaje automático y redes neuronales. El sistema combina la tarjeta de desarrollo “Jetson Nano” (incorporada a una cámara CSI para la adquisición automatizada de imágenes), con los algoritmos necesarios para el reconocimiento de rostros.

El software del sistema se fundamenta en S.O Linux, además del lenguaje de programación Python en el cual se desarrolló el sistema en su totalidad. También se empleó librerías para el tratamiento digital de imágenes como OpenCV, Dlib, entre otras. Para la detección y extracción de características faciales se utilizó redes neuronales pre-entrenadas y para la clasificación y reconocimiento se usaron dos tipos de algoritmos de aprendizaje automático.

Para evitar suplantaciones de identidad se incorpora un método de activación, mediante detección de tono que trae consigo un sistema de registro de ingreso cuando una persona es autenticada correctamente, el sistema está desarrollado en un ‘LAMP’ (combinación de software), donde se inscribe el nombre de la persona, la fecha y hora de ingreso.

Los resultados muestran que para los intereses del proyecto el método de aprendizaje automático KNN es la mejor técnica de clasificación de rostros destacando en dos de las tres pruebas aplicadas.

Palabras clave:

- **RECONOCIMIENTO FACIAL**
- **JETSON NANO**
- **ALGORITMO DE APRENDIZAJE AUTOMÁTICO**
- **REDES NEURONALES PRE-ENTRENADAS**
- **CONTROL DE ACCESO**

Abstract

This project presents the design and implementation of a non-intrusive biometric access control system based on facial recognition using machine learning algorithms and neural networks. The system combines the Jetson Nano development card (incorporated into a CSI camera for automated image acquisition) with the algorithms necessary for face recognition.

The system software is based on S.O Linux, in addition to the Python programming language in which the entire system is located. Libraries were also used for digital image processing such as OpenCV, Dlib, among others. Pre-trained neural networks were detected for the detection and extraction of facial features, and two different types of machine learning algorithms are used for classification and recognition.

To avoid identity theft, an activation method is incorporated, by means of tone detection that brings with it an entry registration system when a person is authenticated correctly, the system is developed in a 'LAMP' (software combination), where it registers the name of the person, the date and time of admission.

The results show that for the interests of the project the KNN machine learning method is the best face classification technique standing out in two of the three tests applied.

Key words:

- **FACIAL RECOGNITION**
- **JETSON NANO**
- **AUTOMATIC LEARNING ALGORITHM**
- **PRE-TRAINED NEURAL NETWORKS**
- **ACCESS CONTROL**

Capítulo 1

Introducción

Antecedentes

Tras la aparición de computadoras en la década de los 50's, se origina al interior de la comunidad científica la interrogante relacionada con la posibilidad de enseñar a estas máquinas a realizar tareas generalmente asociadas con la inteligencia humana, entre las cuales se encuentra la capacidad de resolver problemas, comprender lenguajes o analizar información visual (Sanabria, 2011).

Dando origen con ello a una disciplina orientada a emular la inteligencia humana, denominada "inteligencia artificial" y una miríada de aplicaciones y campos de investigación científica. Uno de los subcampos que más sobresale es la visión artificial, la cual se encarga del análisis de información visual, siendo su insumo básico de entrada una imagen obtenida mediante una cámara (Ranz, 2008).

La visión artificial ha evolucionado logrando avances importantes en distintas aplicaciones tecnológicas como reconocimiento de placas de vehículos (Draghici, 1997), bioprocesos de alimentos (Cubero, 2012), mejora de procesos (Constante Prócel, 2016), sistemas de navegación inteligente (Kumar, 2017), reconocimiento facial (Raducanu & Vitria, 2008), solo por nombrar algunas.

Haciendo énfasis en la última, la investigación en reconocimiento facial automático comenzó en la década de 1960 con el trabajo innovador de Bledsoe, el cual propuso el primer sistema semiautomático para el reconocimiento facial en el que el administrador primero tenía que localizar características como la nariz, la boca, los ojos y las orejas en las fotografías y luego calcular las distancias y las proporciones a un punto de referencia común (Bledsoe, 1968).

Para la década de los 70's Goldstein, Harmon y Lesk introdujeron el uso de 21 marcadores específicos como el color del cabello, los lóbulos de los ojos, el grosor de los labios, etc. para realizar el reconocimiento (Rajeshwar, Ritu, & Dharmender, 2012).

La implementación del primer sistema de reconocimiento facial automático totalmente funcional se produjo en 1988, cuando Kirby y Sirovich aplicaron una técnica estándar de álgebra lineal llamada análisis de componentes principales (PCA). Esta técnica demostró que menos de cien valores son capaces de codificar con precisión una imagen facial adecuadamente alineada y normalizada (Kirby & Sirovich, 1988).

Durante el período de 1993-2000, DARPA y NIST lanzaron el programa FERET (Face Recognition Technology) para alentar el mercado comercial de reconocimiento facial. El programa consistió en la creación de una gran base de datos de imágenes faciales de prueba, con la esperanza de que aquello podría inspirar innovación y generar una tecnología de reconocimiento facial potente (West, 2017).

Para el 2010 las redes sociales le daban la bienvenida al reconocimiento facial, siendo así, que Facebook comenzó a usar una función de reconocimiento facial que ayudaba a detectar personas con rostros destacados en las fotos actualizadas por sus usuarios. Al no tener un impacto negativo, cada día se cargan y etiquetan más de 350 millones de fotos utilizando el reconocimiento facial (Dharaiya, 2020)

Prontamente en el 2011, la convergencia del aprendizaje automático, las redes neuronales y el aprendizaje profundo con el reconocimiento facial avizoró un sinfín de posibles aplicaciones en este campo, agrupándoles en 3 principales categorías: seguridad, salud y comercio.

Actualmente, el área de reconocimiento facial recibe una significativa atención, con un notable aumento en el número de publicaciones. Se han propuesto muchos

enfoques que han llevado a diferentes algoritmos. Pero la mayoría de ellos se centran en 3 aspectos básicos: detección, extracción y reconocimiento (Marqués, 2010).

La *Detección* está definida como el procedimiento que ciertos algoritmos realizan a fin de ubicar o localizar caras en imágenes o videos. De entre los algoritmos destacados, se encuentran: ingenface, análisis de componentes principales (PCA), redes neuronales (NN), discriminante de Fisher y modelos oculto de Márkov (HMM) (Marqués, 2010).

La *Extracción* está definida como el procedimiento para obtener información relevante de un rostro en una imagen, como por ejemplo regiones de la cara, variaciones, ángulos o medidas del rostro. Algoritmos destacados son: histogramas de gradientes orientados (HOG), características similares de HAAR, patrones binarios locales (LBP) y redes neuronales convolucionales (CNN) (Obando, 2019).

El *Reconocimiento* implica un método de comparación, un algoritmo de clasificación y una medida de precisión. Algoritmos de clasificación destacados son: K-vecinos más cercanos (KNN), máquinas de vectores de soporte (SVM), arboles de decisión, regresión logística, descenso de gradiente estocástico, bosque aleatorio (Marqués, 2010).

Justificación e Importancia

Tecnologías disruptivas como es el caso del reconocimiento facial, actualmente están marcando tendencia a nivel mundial por la miríada de aplicaciones en las cuales esta puede ser empleada. Trabajar con este tipo de tecnologías representa un gran desafío y la vez una oportunidad para la investigación y desarrollo del país.

Este tipo de tecnología en el Ecuador ha sido, poco o casi nada aprovechada, no es hasta inicios del anterior año que el ministro de Telecomunicaciones propone 3 aplicaciones de reconocimiento facial como parte de la estrategia Ecuador Digital que

promueve el Gobierno Nacional. Debido al alto potencial aplicativo que ofrece esta tecnología, puede considerarse como un nicho de mercado a explotar.

El reconocimiento facial ha sido una de las áreas significativas e interesantes en el campo de la seguridad y la biometría desde la década de 1990. La importancia de incrementar la seguridad en ciudades y espacios cerrados hacen que la tecnología de identificación y verificación facial sea cada vez más implementada en este nuevo milenio (Cortés, Ardila, & Mendoza, 2011). Ejemplos escalables de esto se ven reflejados en aplicaciones como la búsqueda de delincuentes, búsqueda de personas extraviadas, identificación para el ingreso a organizaciones públicas y privadas, control en aeropuertos, etc.

Además de elevar los niveles de seguridad, el reconocimiento facial permite al usuario conseguir una mayor comodidad en la actividad a realizar, siendo uno de los sistemas menos intrusivos en el control de accesos (Introna & Nissenbaum, 2009)

Con lo anteriormente expuesto, la presente propuesta aspira la elaboración de un sistema de control de acceso biométrico robusto y portable, utilizando algoritmos computacionales que permitan el reconocimiento facial alcanzando evidenciar cuál de los diferentes algoritmos de aprendizaje automático empleados es el más óptimo para este tipo de aplicaciones.

De igual manera, el sistema de reconocimiento facial que se propone tiene la intención de reducir en parte la brecha tecnológica que se tiene con los países desarrollados y transformándose a su vez en un posible emprendimiento.

Alcance del proyecto

Con el proyecto de investigación se espera implementar un sistema de identificación y verificación facial a través de redes neuronales profundas y modelos de aprendizaje automático, que sea eficiente en función del tiempo de procesamiento y porcentaje de acierto conforme el algoritmo de aprendizaje automático implementado.

Se espera portabilidad para el sistema por lo tanto se lo desarrollara en una tarjeta de bajo coste (ej. Raspberry Pi, Orange Pi Zero, etc.) junto con una cámara CSI diurna con una resolución que permita la realización del proyecto propuesto.

Se pretende conocer qué algoritmo de aprendizaje automático resulta ser más eficiente dentro de parámetros de velocidad y precisión, mediante su respectivo análisis comparativo. Las conclusiones obtenidas en el proyecto de investigación, serán presentadas con el fin de ayudar a futuras investigaciones relacionadas con el tema.

Objetivos

General

- Desarrollar e implementar un sistema de control de acceso en tiempo real a través de identificación y verificación facial mediante redes neuronales profundas con distintos algoritmos de aprendizaje automático.

Específicos

- Evaluar dos algoritmos de aprendizaje automático aplicados al reconocimiento de rostros y seleccionar el que mejor rendimiento presenta para su posterior implementación.
- Elaborar un análisis comparativo entre los algoritmos de aprendizaje automático seleccionados en cuanto a precisión y velocidad de respuesta.

- Estructurar una base de datos mediante el registro de información de imágenes y almacenarlas para su posterior procesamiento.
- Diseñar un sistema biométrico robusto que disminuya la probabilidad de una suplantación de identidad.
- Levantar la base de datos de rostros de personas para la implementación del sistema de reconocimiento facial.
- Disponer de una base de datos de personas que utilizaron el sistema de control de acceso a través del registro de información relevante.
- Diseñar un control de acceso integrado al sistema de reconocimiento facial.

Resumen de contenidos

El documento se encuentra estructurado por los siguientes capítulos: el segundo contiene el marco teórico el cual hace referencia a la inteligencia artificial, algoritmos computacionales, y características de la tarjeta de desarrollo empleada. El tercer capítulo representa la explicación sobre el desarrollo e implementación de los dos algoritmos de aprendizaje automático para el reconocimiento facial, así como los algoritmos para la activación de tono y el registro del personal respectivamente. En el cuarto capítulo se presentan las pruebas a las cuales fue sometido el sistema y los resultados obtenidos. Finalmente, en el capítulo cinco se expresa las conclusiones, recomendaciones y trabajos futuros relacionados al proyecto desarrollado.

Capítulo 2

Marco teórico

Introducción a la biometría

Según Bolle (2004) "La biometría es la ciencia de identificar y verificar la identidad de una persona en función de sus características fisiológicas o de comportamiento" (pp. 6), pero a menudo los términos identificación y verificación suelen confundirse, no obstante, son muy diferentes.

La identificación ocurre cuando el sistema intenta responder la pregunta- ¿Quién es X? -, mediante una comparación "uno a muchos" (1: N), a diferencia de la verificación que ocurre cuando el sistema debe responder a la pregunta - ¿Esto es X? - después de que el usuario afirma ser X, mediante una comparación "uno a uno" (1:1) (Woodward, John, Horn, Gatune, & Thomas, 2003)

Como se había manifestado, la biometría puede categorizarse principalmente en dos aspectos:

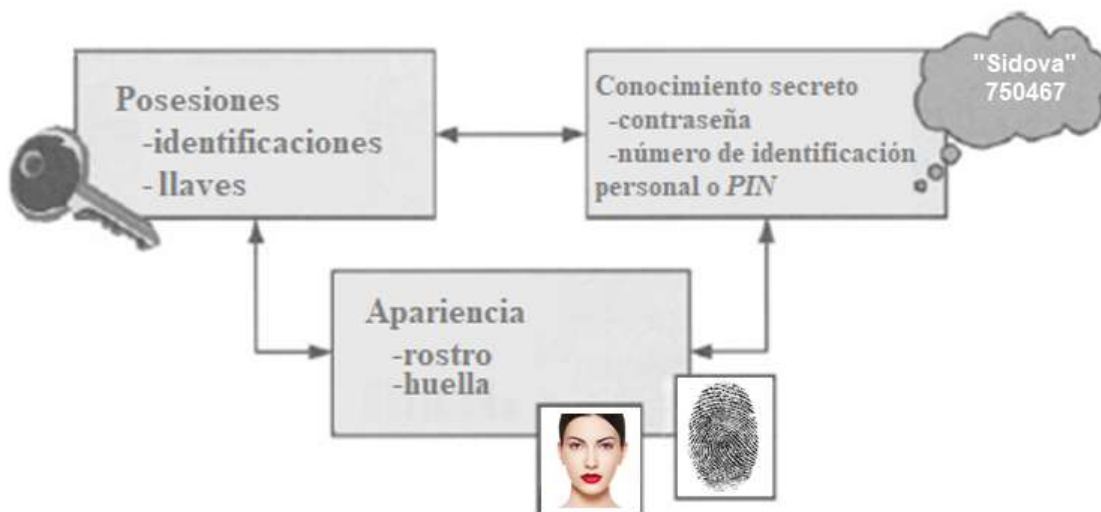
- Características fisiológicas: son rasgos físicos como las huellas digitales, la geometría de la mano, los patrones oculares y los rasgos faciales.
- Características de comportamiento: se consideran aquellas que se basan en la conducta, como la impresión de voz, patrones de escritura e incluso el modo de caminar.

Reconocer a las personas es una actividad fundamental en el corazón de la sociedad y cultura, ya que, para muchas actividades, garantizar la identidad y la autenticidad de las personas es un requisito (Bolle, Connel, Pankanti, & Senior, 2004).

El proceso de verificar la identidad de una persona (autenticación) se fundamenta en un sencillo modelo que postula tres componentes básicos: algo que una persona sabe (un código), o posee (una tarjeta), o tiene (una característica física) (Miller, 1994), La Figura 1 expone lo dicho gráficamente.

Figura 1

Tres formas básicas en que una persona puede demostrar su identidad



Nota. Adaptado de *Guide to Biometrics* (p. 5), por M. Bolle, 2014, Springer Science Business Media LLC.

La identificación biométrica se puede ver como autenticación biométrica "pura" y es mucho más difícil de diseñar e implementar. Los sistemas biométricos deben cumplir con ciertas condiciones biométricas para la identificación de los individuos. (Clarke, 1994), refiere la existencia de un conjunto de características biológicas mediante las cuales se consigue identificar a una persona. Dichas características se presentan en la Tabla 1.

Tabla 1*Características biológicas para la identificación*

Característica	Descripción
Universalidad	Cada persona debe tener un identificador
Unicidad	Cada persona debe tener solo un identificador
Permanencia	El identificador no debe cambiar ni ser modificable
Coleccionable	El identificador debe poder ser medible y cuantificable.
Indispensabilidad	El identificador debe ser una o más características naturales, que cada persona tiene y conserva.

Además de cumplir con dichas condiciones, un sistema biométrico real debe considerar:

- Funcionamiento: el porcentaje de reconocimiento debe superar un determinado umbral y el tiempo de ejecución debe ser asequible.
- Aceptabilidad: debe ser aceptado por las personas como un método de reconocimiento que puedan integrar en su vida diaria.
- Robustez: fortaleza frente ataques fraudulentos (suplantación de identidad)

Actualmente existe un gran número de sistemas biométricos basados en diferentes características físicas. Cada uno de ellos posee ciertas ventajas e inconvenientes, por lo que la elección final debe realizarse en función de la aplicación que se dé al sistema (Vilda, 2006).

Sistemas biométricos

Un sistema biométrico es esencialmente un mecanismo de reconocimiento de patrones que opera adquiriendo datos biométricos de un individuo, extrayendo un conjunto de características de los datos adquiridos y comparando este conjunto de

características con el conjunto de plantillas en la base de datos (Jain, Ross, & Prabhakar, 2004).

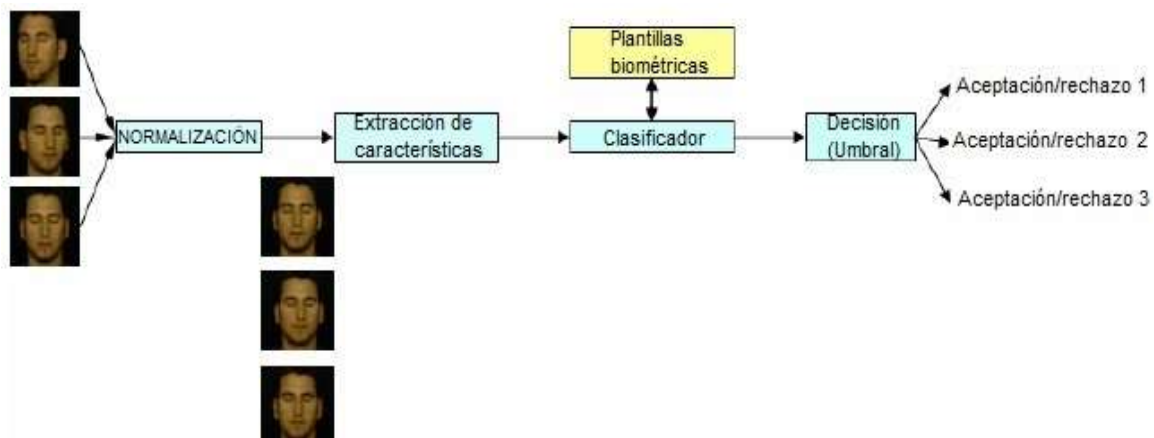
Todo sistema biométrico está compuesto por cuatro módulos principales:

- Módulo de normalización: se caracteriza por modificar los datos biométricos obtenidos para que cumplan condiciones de posición, iluminación, etc.
- Módulo de extracción de características: extrae características discriminatorias de los datos biométricos adquiridos
- Módulo de emparejamiento: genera puntuaciones de coincidencia a través de la comparación entre las características extraídas y las plantillas almacenadas.
- Módulo de decisión: determina la aceptación o rechazo de un individuo dependiendo de la salida del clasificador y el valor umbral.

La Figura 2 representa gráficamente el proceso de cada módulo con el que cuenta un sistema biométrico.

Figura 2

Esquema general de todo sistema biométrico



Nota. Adaptado de *Verificación facial multimodal: 2D y 3D* (p. 13), por C. Conde, 2006.

La Inteligencia artificial (IA)

Es la simulación de procesos de inteligencia humana por parte de máquinas, especialmente sistemas informáticos. Estos procesos incluyen el aprendizaje (la adquisición de información y reglas para el uso de la misma), el razonamiento (empleo de reglas para llegar a conclusiones aproximadas o definitivas) y la autocorrección (resultado final del proceso) (Rouse, 2017).

Aprendizaje profundo

El aprendizaje profundo (Deep-Learning) se refiere a una clase bastante amplia de técnicas y arquitecturas de aprendizaje automático, con el sello distintivo de usar muchas capas de procesamiento de información no lineal que son de naturaleza jerárquica (Yu & Deng, Three Classes of Deep Learning Networks, 2013). En otras palabras, el aprendizaje profundo explota muchas capas de procesamiento de información no lineal para la extracción y transformación de características (supervisadas o no supervisadas), y para el análisis y clasificación de patrones (Arel, Rose, & Karnowski, 2010)

Dependiendo de cómo se pretendan utilizar las arquitecturas y las técnicas, por ejemplo, síntesis/generación o reconocimiento/clasificación, se puede la clasificar en tres clases principales:

- Redes profundas para el aprendizaje no supervisado: se refiere al no uso de información de supervisión específica de la tarea (por ejemplo, etiquetas de clase) en el proceso de aprendizaje.
- Redes profundas para el aprendizaje supervisado: su objetivo es proporcionar directamente el poder discriminatorio para fines de clasificación de patrones.

- Redes profundas híbridas: el objetivo en estas redes es la discriminación mediante modelos de optimización y regularización.

Visión por computador

Comprende métodos y técnicas a través de los cuales, sistemas de visión artificial pueden construirse y emplearse de manera razonable en aplicaciones prácticas (Davies, Twining, & Taylor, 2008).

Básicamente, un sistema de visión por computadora se compone de las siguientes etapas:

- Adquisición de imágenes: inicia por la transferencia de señales electrónicas desde un sensor a una representación numérica por un dispositivo como una cámara (Zareiforoush, Minaei, Alizadeh, & Banakar, 2015). Factor crucial a tomar en cuenta en esta etapa es la calidad de la imagen adquirida.
- Procesamiento de imágenes: se refiere al proceso para diferenciar regiones de interés, con el fin de extraer información. Se puede dividir en procesamiento de bajo nivel, nivel intermedio y alto nivel (Hornberg, 2017).

Actualmente el reconocimiento facial es una de las aplicaciones con mayor demanda como sistema de visión artificial

Reconocimiento facial

El reconocimiento facial se usa para identificar y verificar automáticamente una persona a partir de una imagen (Sekhon & Agarwal, 2015). Una imagen es solo una colección de valores de intensidad de luz y color. El análisis de estos píxeles para la detección de rostros, puede tornarse lento y difícil de lograr debido a las amplias variaciones de forma y pigmentación dentro de un rostro humano (Bradski, 1998).

Pero una de las ventajas de utilizar el rostro, es que se puede emplear la medición geométrica espacial de los puntos clave del rostro, es decir, realizar medición de distancias entre órganos faciales (Woodward, John, Horn, Gatune, & Thomas, 2003).

Actualmente existe gran variedad de métodos para realizar la detección de rostros, cada uno tiene sus propias debilidades y fortalezas. Algunos usan tonos de piel, otros usan contornos y por último hay unos más complejos que usan redes neuronales, filtros y algoritmos de aprendizaje automático. El problema en común de todos los mencionados es el costo computacional que presenta cada uno, siendo directamente proporcional la complejidad del método con el costo computacional.

Aprendizaje automático (ML)

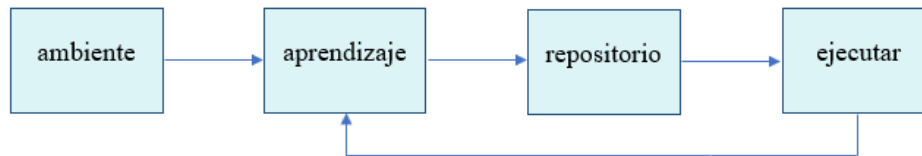
En 1959, se define el aprendizaje de automático como un "campo de estudio que da a las computadoras la capacidad de aprender sin ser programadas de forma explícita" (Arthur, 1959, p. 210).

Partiendo de este hecho, el aprendizaje automático (Machine-Learning, o ML) es una parte de la AI que se encarga de estudiar cómo usar las computadoras para simular actividades de aprendizaje humano y estudiar métodos de auto mejora para obtener nuevos conocimientos y habilidades, identificar conocimientos existentes y mejorar continuamente (Wang, Zhou, & Ma, 2009)

La Figura 3 muestra el modelo básico de ML, el cual tiene como objetivo transformar los datos del medio ambiente en información mediante un proceso de realimentación generando así el verdadero aprendizaje.

Figura 3

Modelo básico de machine learning



Nota. Adaptado de “*A Brief Review of Machine Learning and its Application*” (p. 9), por H. Wuang, 2009, IEEE Xplore.

El proceso de la figura 3 puede también ser representado mediante algoritmos de ML, que son instrucciones matemáticas que utilizan métodos computacionales para aprender la información sin necesidad de seguir una ecuación como modelo predeterminado. Generalmente los algoritmos de aprendizaje automático se los clasifica de 2 maneras: algoritmos por estilo y algoritmos por similitud de función.

En los algoritmos por estilo se encuentran los: supervisados, semi-supervisados y no supervisados.

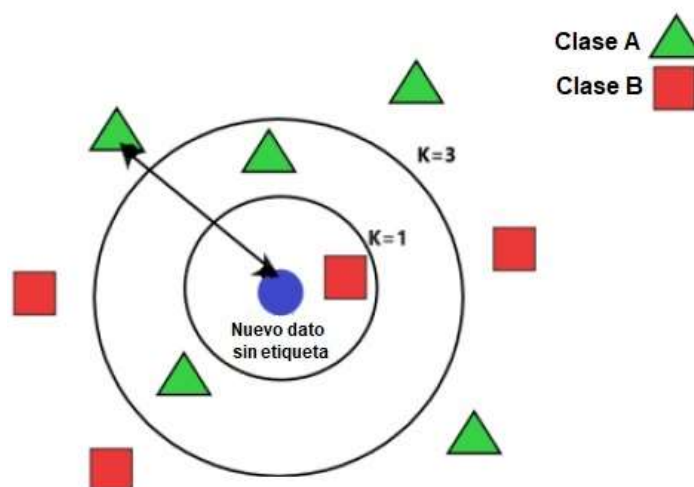
- Algoritmos supervisados: caracterizados porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado (supervisor, maestro) que determina la respuesta del algoritmo a partir de una entrada determinada (datos totalmente etiquetados) (Matich, 2001).
- Algoritmos semi-supervisados: su proceso de aprendizaje se basa en una combinación de datos etiquetados (en menor porción) y datos no etiquetados.
- Algoritmos No supervisados: su proceso de aprendizaje consiste en la clasificación de similitudes y/o diferencias con el resto de datos sin control alguno de un agente externo. (Castro, 2018)

K- vecinos más cercanos (KNN)

Es un algoritmo de aprendizaje automático supervisado, el cual se basa en el principio de que las instancias dentro de un conjunto de datos generalmente existirán cerca de otras instancias que tienen propiedades similares (Cover & Hart, 1967). Si las instancias están etiquetadas con una etiqueta de clasificación, entonces el valor de la etiqueta de una instancia no clasificada se puede determinar observando la clase de sus vecinos más cercanos, como se presenta la Figura 4.

Figura 4

Método de clasificación KNN



Nota. Adaptado de "A Brief Review of Nearest Neighbor Algorithm for Learning and Classification" (p. 1256), por Taunk, K., Sanjukt, D. Verma, S., 2019, IEEE Xplore

KNN localiza las K instancias más cercanas a la instancia de consulta y determina su clase identificando la etiqueta de clase más frecuente (Kotsiantis, 2007)

KNN principalmente es utilizado como clasificador, su mecanismo de funcionamiento conlleva dos etapas:

- Aprendizaje: al algoritmo se le proporciona un conjunto de datos de entrenamiento etiquetado donde los puntos de datos se clasifican en varias clases. Además, se le especifica un valor de K , el cual representa el número de vecinos más cercanos (sesgo) que puede tomar el nuevo dato de entrada (dato sin etiqueta).
- Evaluación: para cada dato sin etiquetar, se calcula la distancia euclidiana con todos los puntos de datos de entrenamiento utilizando la ecuación 1:

$$\sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (1)$$

Encontrando los K -vecinos más cercanos. Luego el algoritmo determina a que clase pertenece el nuevo dato según el número máximo de vecinos más cercanos.

Los clasificadores basados en KNN han sido aplicados en el reconocimiento facial, por ejemplo, en (Lucas, 1997), que tuvo un rendimiento muy alto usando un $K=1$ y distancia no euclidiana. De igual manera en (Belhumeur, 1997) utiliza el clasificador KNN, sin embargo, experimenta un menor rendimiento debido a las variaciones de iluminación o de gesto en el conjunto de prueba.

Máquina de vectores de soporte (SVM)

Las máquinas de soporte vectorial (Support Vector Machine) son un método de aprendizaje supervisado que genera funciones de clasificación a partir de un conjunto de datos de entrenamiento etiquetados (Cortés, Ardila, & Mendoza, 2011).

El aprendizaje de las SVMs se basa en la construcción de un Hiperplano de Separación Óptimo (HSO) cuya posición se calcula entre una familia de hiperplanos de la forma expresada en ecuación 2:

$$\{\vec{w} \cdot \vec{x} + b = 0\}, \vec{w} \in \mathfrak{R}^n, \vec{x} \in \mathfrak{R}^n, b \in \mathfrak{R} \quad (2)$$

Donde:

\vec{w} : es el vector de pesos

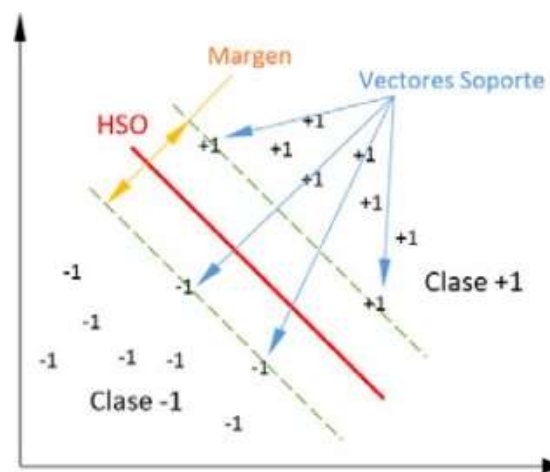
\vec{x} : es el vector de entrada

b : es un bias

Observando la Figura 5 y de acuerdo a la notación anterior el hiperplano de separación óptimo se encuentra definido por el máximo margen de separación entre las dos clases (González, Barrientos, Toapanta, & Del Cerro, 2017).

Figura 5

Hiperplano de Separación Óptimo implementado para dos clases

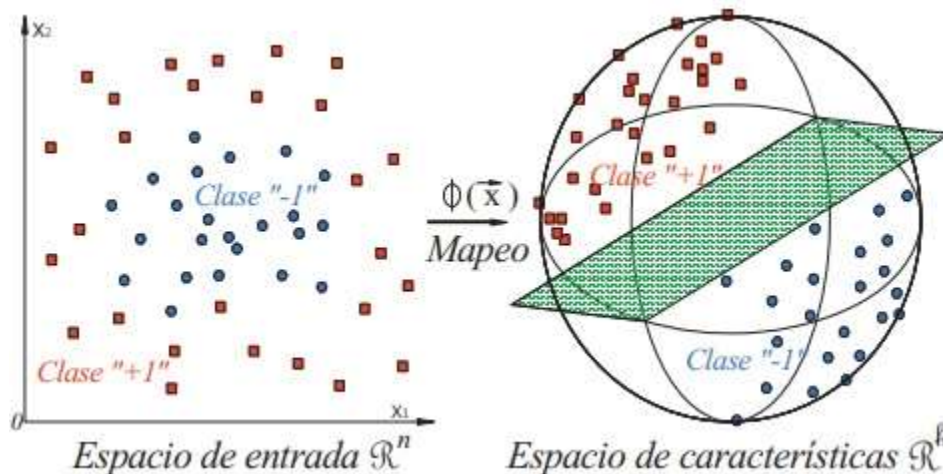


Nota. Adaptado de “Aplicación de las Máquinas de Soporte Vectorial (SVM) al diagnóstico clínico de la Enfermedad de Pákinson y el Temblor Esencial” (p.398), por R. González & otros, 2017, Revista Iberoamericana de Automática e Informática Industria

Cuando las clases no pueden ser separadas linealmente como en el de la Figura 6, se procede a mapear los datos de entrada a una dimensión superior $\mathfrak{R}^n \rightarrow \mathfrak{R}^h$, mediante la utilización kernels $K(\vec{x}_i, \vec{x}_k) = \varphi(\vec{x}_i) \cdot \varphi(\vec{x}_k)$, que no son más que funciones matemáticas (Statnikov, Aliferis, Hardin, & Guyon, 2011).

Figura 6

Mapeo de datos a un espacio de mayor dimensión



Nota. Adaptado de “Aplicación de las Máquinas de Soporte Vectorial (SVM) al diagnóstico clínico de la Enfermedad de Párkinson y el Temblor Esencial” (p.399), por R. González & otros, 2017, Revista Iberoamericana de Automática e Informática Industria

Redes neuronales

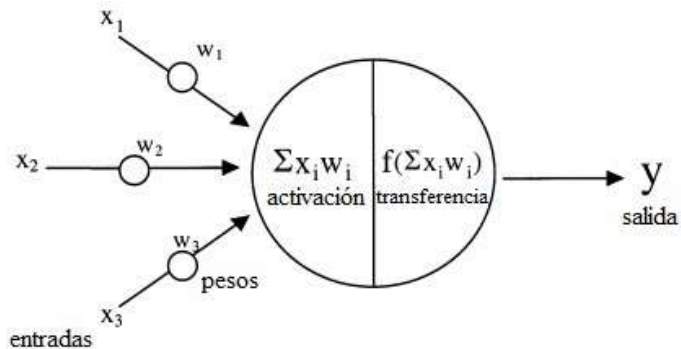
Red neuronal artificial

Una Red Neuronal Artificial (ANN: Artificial Neuronal Network), es un modelo computacional de inspiración biológica formado por cientos de unidades individuales, neuronas artificiales, conectadas con coeficientes (pesos) que constituyen la estructura neural (Agatonovic & Beresford, 2000).

Esto se traduce a que, las señales que llegan (entradas) son multiplicadas por los pesos de conexión realizando una sumatoria para luego pasar por una función de transferencia y producir la salida para esa neurona, la Figura 7 ilustra lo mencionado.

Figura 7

Modelo de una neurona artificial



Nota. Adaptado de "Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research" (p. 719), por S. Agatonovic, 2000, Elsevier Science B.V., 22.

En términos matemáticos se puede traducir mediante la ecuación 3:

$$y_i = f_i \left(\sum_{j=1}^n w_{ij} x_j - \theta_i \right) \quad (3)$$

Donde:

i son los nodos

f_i es la función de transferencia (activación)

y_i es la salida del nodo

x_j es la entrada j th para el nodo

w_{ij} es el peso que conecta cada nodo

θ_i es el umbral (o sesgo) del nodo

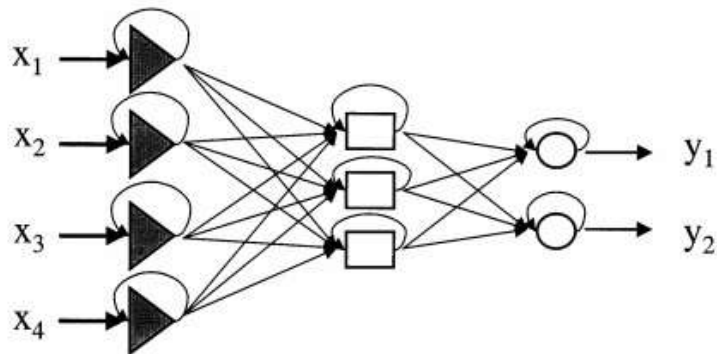
Normalmente, f_i es una función no lineal, como las funciones: *sigmoide*, *heaviside* o *Gausiana*. La ecuación (3) solamente involucra una entrada (x_j), relacionada con el nodo de la red neuronal. Para una red neuronal de orden superior, es decir, en donde los nodos tienen más de una entrada involucrada (Yao, 1999), se utiliza la ecuación 4:

$$y_i = f_i \left(\sum_{j,k=1}^n w_{ijk} x_j x_k - \theta_i \right) \quad (4)$$

Hay muchos tipos de redes neuronales, pero generalmente estas pueden ser descritas por la función de transferencia de sus neuronas, por su proceso de aprendizaje y por su conexión (Yu & Deng, 2013).

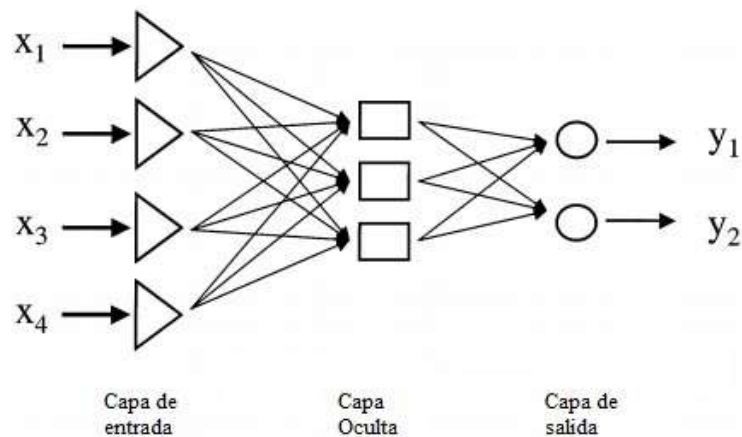
- Por su proceso de aprendizaje: consiste en ir ajustando iterativamente los pesos de conexión, este proceso se lo puede dividir en tres tipos: aprendizaje supervisado, no supervisado y aprendizaje reforzado.
- Por su conexión: es la manera en que las neuronas permanecen conectadas unas a otras, generando un gran impacto en la red artificial. Se tiene dos tipos de conexión: feedback y feedforward.

Las redes Feedback tienen conexiones desde las neuronas de salida a las de entrada; esta, guarda un recuerdo del estado anterior para que el siguiente estado no solo dependa de las señales de entrada sino también de los estados anteriores de la red (retroalimentación), una representación gráfica se muestra en la figura 8 (Agatonovic & Beresford, 2000).

Figura 8*Red neuronal feedback*

Nota. Adaptado de “*Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research*” (p. 720), por S. Agatonovic, 2000, Elsevier Science B.V., 22.

Las redes FeedForward, como se observa en la Figura 9, no tiene conexiones de realimentación y por lo tanto no mantiene un registro de sus valores de salida anteriores.

Figura 9*Red neuronal feedforward*

Nota. Adaptado de “*Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research*” (p. 720), por S. Agatonovic, 2000, Elsevier Science B.V., 22.

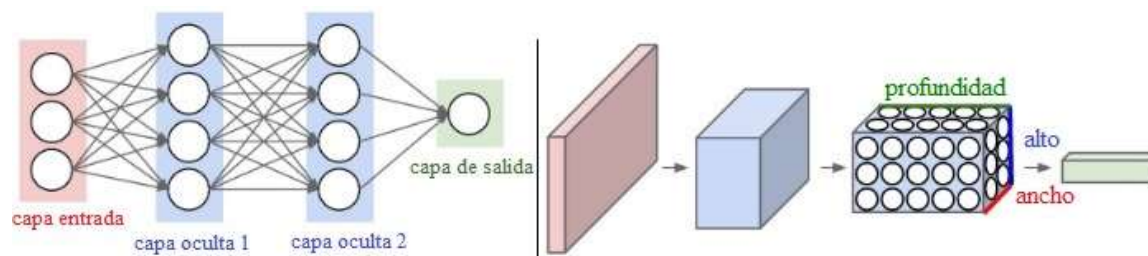
Red neuronal convolucional (ConvNet)

Es un tipo de red artificial que utiliza la convolución como una de sus principales herramientas para el procesamiento y análisis de datos, más concretamente de imágenes. Se utiliza para la clasificación y agrupamiento de imágenes, así como para el reconocimiento de objetos y de rostros en las mismas.

Las redes neuronales convolucionales o ConvNets, aprovechan el hecho de que la entrada consiste en imágenes y limitan la arquitectura de una manera más sensata. Como se puede ver en la Figura 10, a diferencia de una red neuronal normal, las capas de un ConvNet tienen neuronas dispuestas en 3 dimensiones: ancho, alto, profundidad (Saha, 2018).

Figura 10

Contraste de red neuronal normal vs red neuronal convolucional

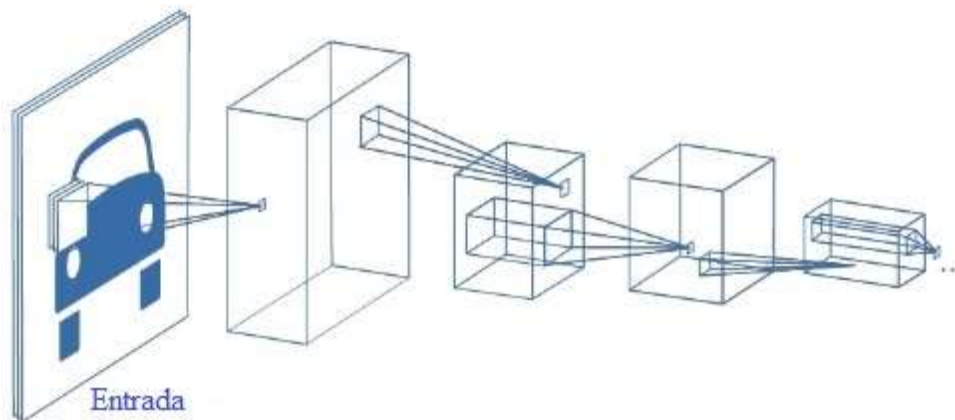


Nota. Adoptado de Architecture Overview, por CS231n Convolutional Neural Network for Visual Recognition (<https://cs231n.github.io/convolutional-networks/#overview>)

Otra de las particularidades en una ConvNet es que las neuronas de una capa solo se conectarán a una pequeña región de la capa anterior, en lugar de todas las neuronas de una manera completamente conectada como se aprecia en la Figura 11.

Figura 11

Conexión entre capas ConvNet



Nota. Adaptado de A Comprehensive Guide to Convolutional Neural Networks, 2018, por Towardsdatascience (<https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>)

De la Figura 11 se observa que la capa de salida final tendría dimensiones $1 \times 1 \times \text{profundidad}$, porque al final de la arquitectura ConvNet se reduce la imagen completa en un solo vector de puntajes de clase, dispuestos a lo largo de la dimensión de profundidad (Saha, 2018).

Una ConvNet está comprendida principalmente por tres tipos de capas neuronales: capas convolucionales, capas de agrupación y capas completamente conectadas. Cada tipo de capa juega un papel diferente como se observa en la Figura 12.

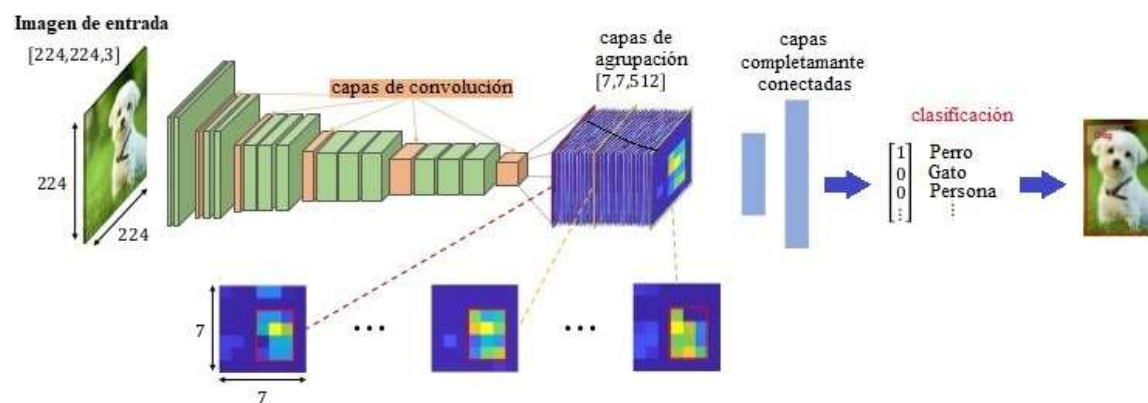
- Capas Convolucionales: en estas capas la ConvNet utiliza varios kernels (filtros) para convolucionar toda la imagen de entrada generando mapas de características (Voulodimos & Doulamis, 2018).
- Capas de Agrupación: se encargan de reducir las dimensiones espaciales (ancho \times altura) del volumen de entrada para la siguiente capa convolucional sin afectar

la dimensión de profundidad. La operación realizada por esta capa también se denomina submuestreo o disminución de resolución, ya que la reducción del tamaño conduce a una pérdida simultánea de información; pérdida beneficiosa en términos de sobrecarga computacional. La agrupación promedio y la agrupación máxima son las estrategias más utilizadas (Krizhevsky, Sutskever, & Hinton, 2012).

- Capas completamente Conectadas: en esta zona se da el razonamiento de alto nivel en la red neuronal. Las neuronas en una capa totalmente conectada tienen conexiones completas a toda la activación en la capa anterior. Por lo tanto, su activación se puede calcular con una multiplicación de matriz seguida de un desplazamiento de sesgo. Las capas completamente conectadas eventualmente convierten los mapas de características 2D en un vector de características 1D. El vector derivado podría ser alimentado hacia un cierto número de categorías para su clasificación o podría considerarse como un vector de características para su posterior procesamiento (Girshick, Donahue, Darrell, & Malik, 2014).

Figura 12

Arquitectura de una ConvNet en la detección de un objeto



Nota. Adaptado de “Object Detection: Localization and classification” (p. 3), por H. kim, 2018, Mediviewsoft, Department of computational science and engineering

Software y hardware

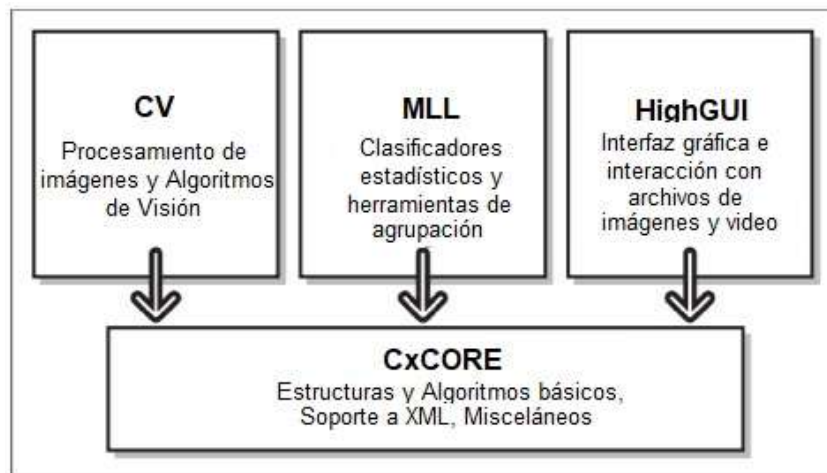
OpenCV

Open Source Computer Vision, es una librería con más de 500 algoritmos optimizados para procesamiento de imágenes y video, además, incluye una librería de MLL para clustering y reconocimiento de patrones (Mordvintsev, 2017). Se encuentra disponible para la mayoría de las plataformas, Microsoft, Linux, OS X, entre otras.

OpenCV fue diseñado para la eficiencia computacional y con un fuerte enfoque en las aplicaciones en tiempo real (Datal & Patel, 2013). Fundamentalmente está constituida por 4 elementos claves como se muestra en la Figura 13.

Figura 13

Arquitectura de OpenCV



Nota. Adaptado de “*Optimization of Image Processing Algorithms on Mobile Platforms*” (p.4), por M. Shirvaikar, 2011, Proceedings of SPIE.

Python

Es un lenguaje de programación de fácil aprendizaje. Tiene estructuras de datos eficientes de alto nivel y un enfoque simple pero efectivo para la programación orientada a objetos. Además, integra una semántica dinámica y sintaxis elegante para el desarrollo

web y de aplicaciones. Tanto el intérprete como las librerías estándar se encuentran disponibles de forma gratuita en el sitio web oficial de Python (Van Rossum, 2017).

Face recognition

Es una biblioteca de reconocimiento facial de código abierto, está fundamentada en el kit de herramientas de Dlib la cual está provista de algoritmos de: aprendizaje automático, aprendizaje profundo, procesamiento de imágenes, redes neuronales convolucionales, predictores de forma entre otros. (Dlib, 2020)

La biblioteca esta entrenada en el conjunto de datos LFW (Labelled Faces in the Wild) presumiendo una precisión de acierto del 99%.

Jetson nano

Es una tarjeta de desarrollo de NVIDIA de gran alcance que permite ejecutar software de inteligencia artificial, como redes neuronales y algoritmos de aprendizaje automático en paralelo para aplicaciones como clasificación de imágenes, detección de objetos, segmentación, procesamiento del habla, reconocimiento facial entre otras (B. Lewis, 2019). En la Figura 14 se muestra el módulo físicamente.

Figura 14

Módulo Jetson nano



Nota. Adaptado de JetsonNano-DevKit_Front-Top_Right_trimmed por NVIDIA, Jetson Nano Developer Kit (<https://developer.nvidia.com/embedded/jetson-nano-developer-kit>)

Cámara CSI

Es un periférico destinado a conectarse con la interfaz serial provista en la tarjeta de desarrollo, la cámara diurna de raspberry pi versión 2.0 seleccionada, brinda una transferencia de datos de 800 megabytes por segundo (MB /s) y una resolución de 3280 × 2464 píxeles. En la Figura 15 se muestra un ejemplar.

Figura 15

Cámara raspberry pi V2



Nota. Adaptado de V2 Camera Module, CSI-2, 3280 x 2464 Resolution por Raspberry Pi, Camera module (<https://www.raspberrypi.org/documentation/hardware/camera/>)

Capítulo 3

Diseño e implementación

Descripción

En este capítulo se describe el diseño de un Sistema de Control de Acceso en tiempo real a través de identificación y verificación facial mediante procesamiento digital de imágenes e inteligencia artificial. En el diseño del sistema se contempla el uso de dos algoritmos de aprendizaje en el módulo de reconocimiento facial para evaluar su desempeño e implementar el que mejor rendimiento presente.

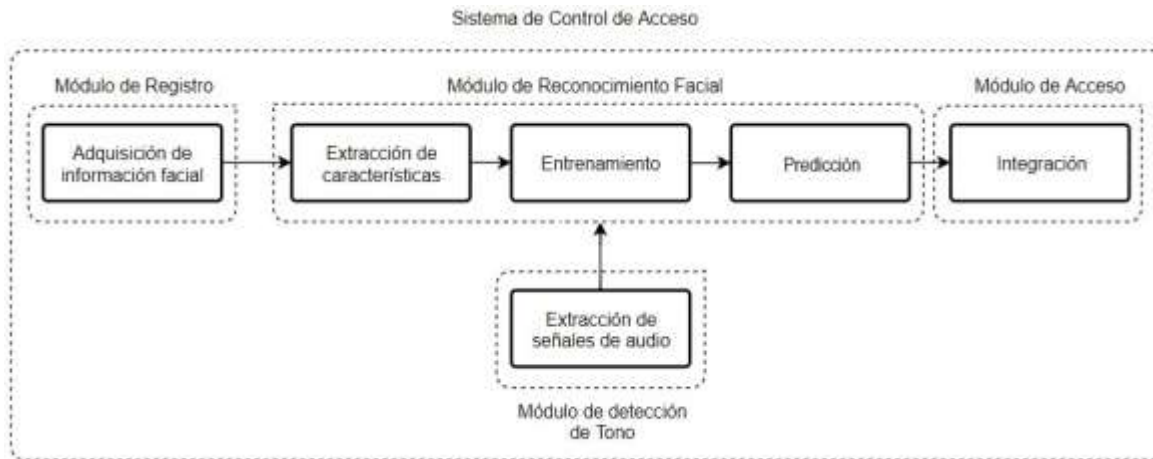
El sistema está conformado por cuatro módulos principales: detección de tono, registro, reconocimiento facial y acceso; de los cuales, dos operan de manera sistemática sobre información previamente recopilada por uno de ellos.

El primer módulo permite el funcionamiento del sistema, emplea técnicas de detección y extracción de señales de audio. El segundo módulo se encarga de recopilar información de rostros empleando técnicas de aprendizaje automático y de procesamiento de imágenes. El tercer módulo es la esencia del proyecto, está encargado del tratamiento de los datos provistos por el módulo anterior y de la generación de nueva información para el módulo siguiente. Emplea técnicas para el tratamiento de imágenes, algoritmos de pérdida, redes neuronales pre-entrenadas, algoritmos de aprendizaje profundo, algoritmos de detección, algoritmos de aprendizaje automático y algoritmos clasificadores. El cuarto y último módulo es el responsable de otorgarle una aplicación al proyecto; en este tienen lugar el manejo de señales de activación y el registro de personas reconocidas, mediante el motor de bases de datos MySQL.

Conforme avance el capítulo, se explicará el diseño de cada módulo del sistema, detallando las técnicas implementadas y la tecnología utilizada. El esquema de la Figura 16 describe el diseño general del sistema.

Figura 16

Diagrama de bloques del sistema de control de acceso



Módulo de detección de tono

El presente módulo permite la activación del módulo de reconocimiento facial mediante la detección de una señal de audio en específico a través del micrófono del sistema. Los pasos que sigue el módulo se presentan en la Figura 17.

Figura 17

Diagrama de componentes del programa *bell.py*



Para lo mencionado se realizó el programa “bell.py” en Python, las bibliotecas necesarias para la detección de señales de audio se presentan en la Tabla 2.

Tabla 2

Librerías para la detección de tono

Librería	Función
Aubio	Permite la extracción de señales de audio
PyAudio	Permite la grabación de audio

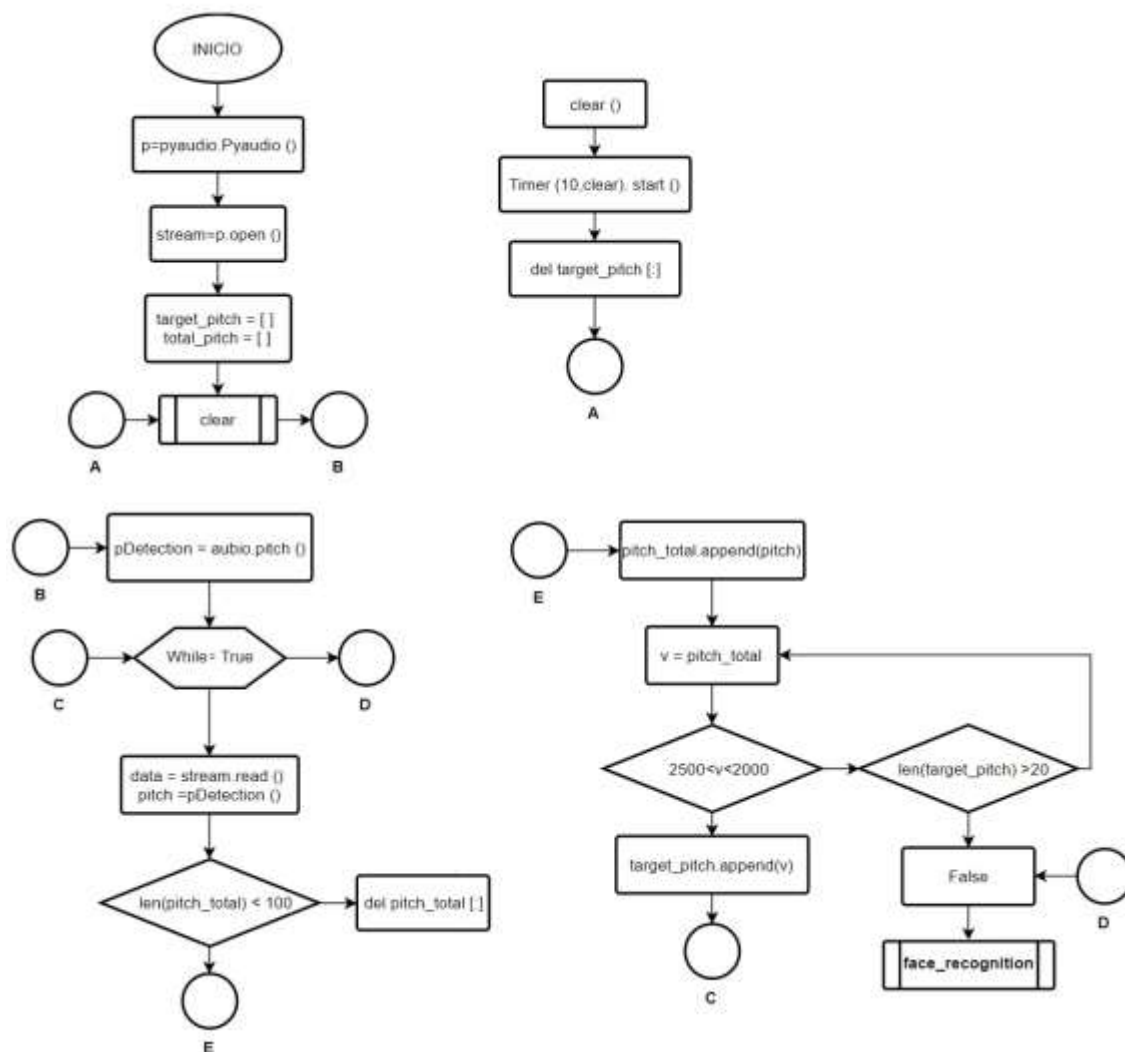
Se requiere establecer un rango determinado de señales de audio a ser detectadas, según algunos estudios realizados el rango de frecuencias fundamentales para voces masculinas se encuentra entre 77Hz a 482Hz con una media de 106 Hz y para voces femeninas se sitúa entre 137 Hz a 634 Hz con una media de 193 Hz. (Paolini, Hernández, & Pereyra, 2018).

Por lo tanto, para evitar falsos positivos, la detección de tono se estableció en un rango mayor a 2KHz, situándose fuera del rango de frecuencia fundamental media tanto para voces masculinas como femeninas.

El programa actualiza la información detectada cada 10 segundos, para que el programa active el módulo de reconocimiento facial, la señal de audio deberá ser detectada 20 veces (3 segundos aproximadamente). En el gráfico de la Figura 18 se presenta el diagrama de flujo del programa.

Figura 18

Diagrama de flujo del programa bell.py



Módulo de registro

Este módulo permite la recopilación y almacenamiento digital de información de rostros mediante técnicas de procesamiento de imágenes y video, en conjunto con modelos de aprendizaje automático. Todo esto se encuentra contenido en un programa elaborado en el lenguaje de programación *Python*; sus resultados serán aprovechados por el siguiente módulo.

Requerimiento tecnológico

Los componentes tecnológicos utilizados para este módulo, tanto de hardware como de software se describen en la Tabla 3. Cabe recalcar que las características detalladas son las mínimas recomendadas.

Tabla 3

Características de los materiales empleados en el registro de usuarios

Software / Hardware	Función	Características Mínimas
S.O. Linux	Sistema operativo de la Pc	Ubuntu 16.04 LTS
Python	Lenguaje de programación	Versión 3.7.3
OpenCV	Librería para el tratamiento de imágenes	Versión 4.1.0
Laptop	Procesamiento del programa	Procesador 64bits, Corei5, 2.6GHz
Webcam	Captura de imágenes en diferentes ángulos	Resolución 1280x720, 17 FPS, color RGB, Bitrate 2.36 MB/s

Adquisición de datos

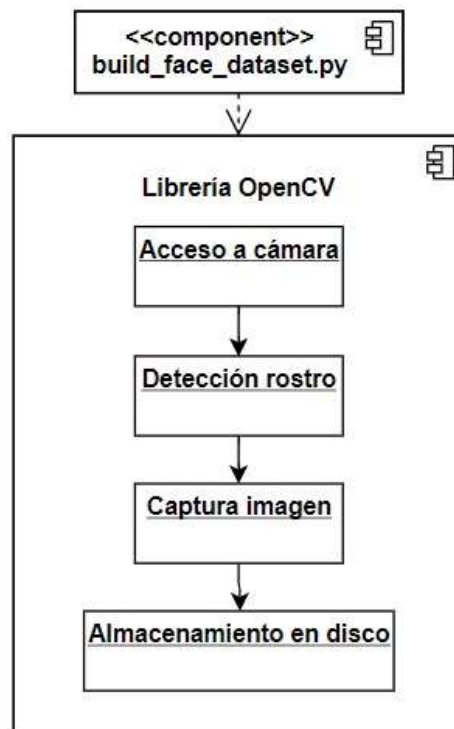
Representa la elaboración de una base de datos, mediante el registro de información fotográfica del rostro de personas. La base de datos (BD) está constituida de

10 personas con un mínimo de 30 fotografías por persona. La obtención de los datos se llevó a cabo en un lugar cerrado con iluminación artificial.

El programa “build_face_dataset.py” aloja la programación para obtención de dicha base de datos. Elementos que destacan en la codificación son *OpenCV* como librería principal para el procesamiento de imágenes y *un clasificador en cascada* como modelo pre-entrenado para la detección de rostros. La Figura 19 muestra un esquema de los componentes del programa.

Figura 19

Diagrama de componentes del programa build_face_dataset.py



Acceso a la cámara

Se procede a la configuración de la cámara web mediante la importación de la librería *imutils* y la creación de un objeto el cual permitirá la inicialización de dicha cámara.

Para obtener una grabación de video se debe mantener una lectura de frames constante por tal razón el objeto creado se encuentra contenido dentro de una estructura repetitiva.

Detección de rostros

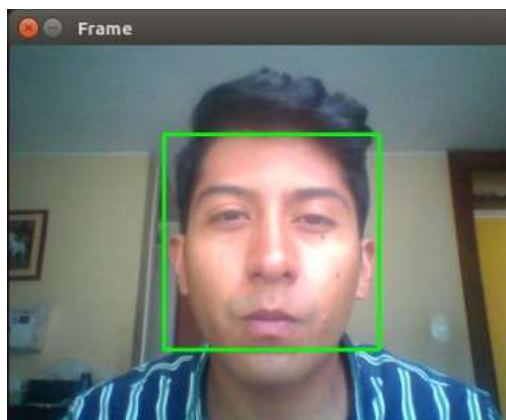
Se carga el modelo pre-entrenado *haarcascade_frontalface_default.xml* desde el disco; este modelo es un algoritmo clasificador de detección de objetos, fundamentado en ML (machine learning), que permite la identificación de rostros en una imagen o vídeo y se basa en el concepto de características Haar (Viola & Jones, 2001).

Para cargar el modelo se crea una instancia de la clase *cv2.CascadeClassifier* (propia de OpenCV), en la cual se especifica dicho modelo está contenido en un archivo XML.

Una vez cargado el modelo, se procede a su aplicación mediante la creación de un objeto, especificándole los frames capturados en video y la escala, lo que permitirá la detección del rostro enfocado. Un recuadro verde en el rostro se podrá visualizar en el video como resultado de la detección, la Figura 20 muestra un ejemplo.

Figura 20

Detección de rostro mediante OpenCV y modelo de Haar



Captura y almacenamiento de imágenes

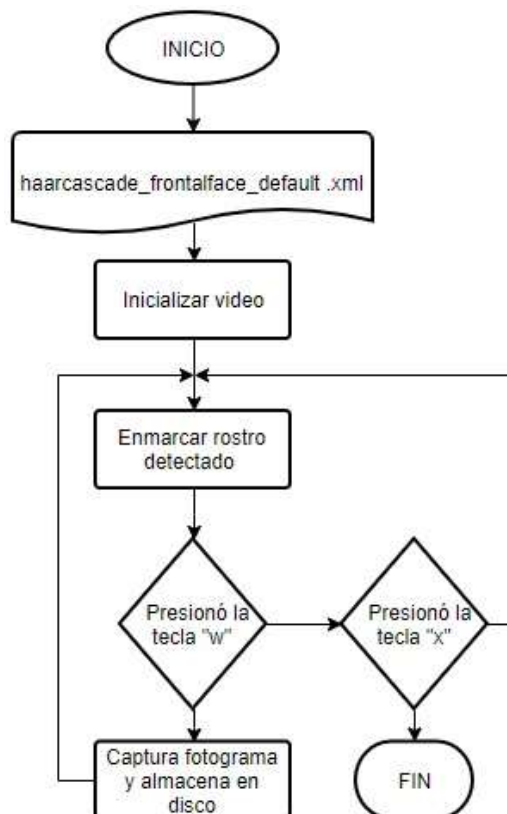
La captura y almacenamiento de la imagen se efectúa al presionar la tecla w; una vez que la persona se ubica en la estación de trabajo y su rostro sea detectado.

Para llevar a cabo dicho proceso se crea un objeto de escritura en el disco, al cual se le proporciona correctamente la dirección de almacenamiento de la información capturada. Los comandos utilizados en ambos escenarios son propios de la librería OpenCV. La codificación de este proceso al estar necesariamente ligado con el de video se encuentran contenidas en la misma estructura repetitiva.

A continuación, para una mejor comprensión del módulo de registro; en la Figura 21 se presenta el diagrama de flujo que obedece el programa.

Figura 21

Diagrama de flujo del programa built_face_dataset.py



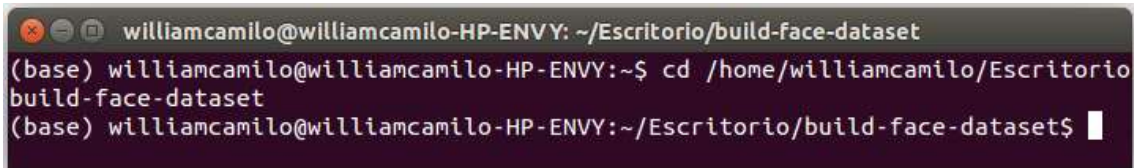
Procedimiento para el registro de rostros

El procedimiento que se explica continuación es ejecutado por cada una de las personas integrantes de la BD.

Como primer paso, la persona debe ubicarse en la estación de trabajo donde se encuentra el pc con la cámara web para proceder con el registro fotográfico. Seguidamente la persona a registrarse debe crear una carpeta con su nombre en un directorio en específico, como se muestra en la Figura 22.

Figura 22

Ruta donde se almacenarán las imágenes (base de datos)

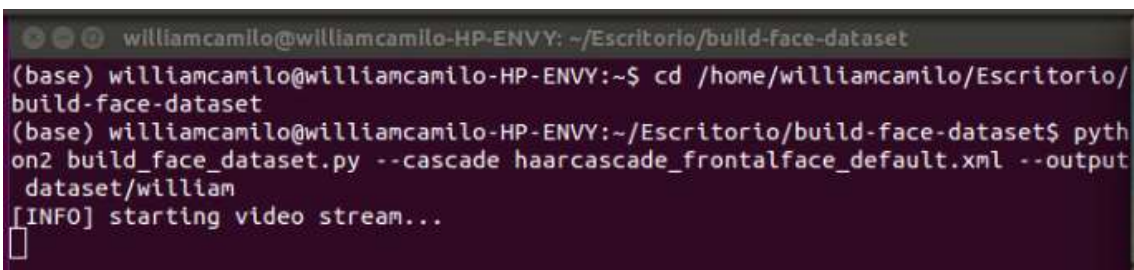
A terminal window with a dark background and light text. The title bar reads "williamcamilo@williamcamilo-HP-ENVY: ~/Escritorio/build-face-dataset". The terminal shows the following commands and output:

```
(base) williamcamilo@williamcamilo-HP-ENVY:~$ cd /home/williamcamilo/Escritorio/build-face-dataset
(base) williamcamilo@williamcamilo-HP-ENVY:~/Escritorio/build-face-dataset$
```

Posteriormente se ejecuta el programa “*build_face_dataset.py*”, especificando la ruta de la carpeta en donde se almacenarán las capturas del rostro de la persona, tal como se muestra en la Figura 23.

Figura 23

*Ejecución del programa *build_face_dataset.py**

A terminal window with a dark background and light text. The title bar reads "williamcamilo@williamcamilo-HP-ENVY: ~/Escritorio/bulld-face-dataset". The terminal shows the following commands and output:

```
(base) williamcamilo@williamcamilo-HP-ENVY:~$ cd /home/williamcamilo/Escritorio/build-face-dataset
(base) williamcamilo@williamcamilo-HP-ENVY:~/Escritorio/build-face-dataset$ python2 build_face_dataset.py --cascade haarcascade_frontalface_default.xml --output dataset/william
[INFO] starting video stream...
█
```

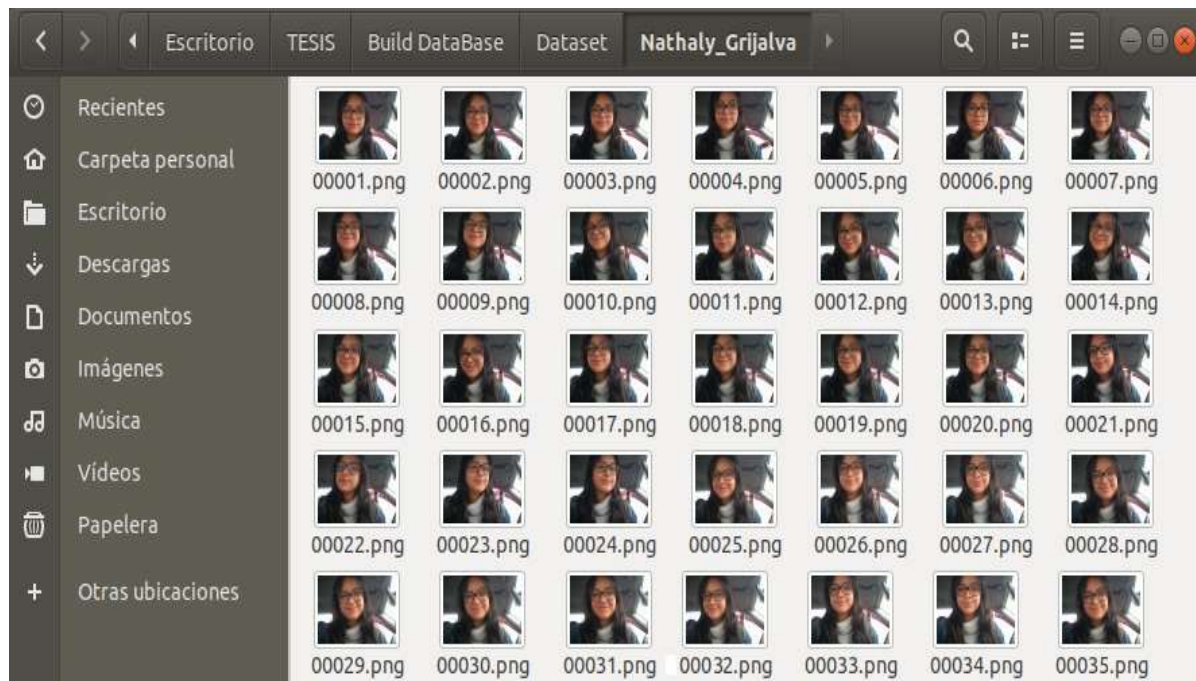
Una vez ejecutado el programa, se desplegará una ventana en donde se inicializa el video, aquí la cámara detectará un rostro y lo enmarcará, cuando esto ocurra, la

persona tendrá que presionar la tecla “w” para que la webcam realice un fotograma y este se almacene automáticamente en la ruta especificada.

La persona podrá realizar la cantidad de fotogramas que desee; para culminar con el procedimiento de registro, la persona deberá presionar la tecla “esc” y se finalizará la grabación de video. La Figura 24 muestra el resultado final del procedimiento llevado a cabo.

Figura 24

Base de datos creada con un integrante de la BD



Módulo de reconocimiento facial

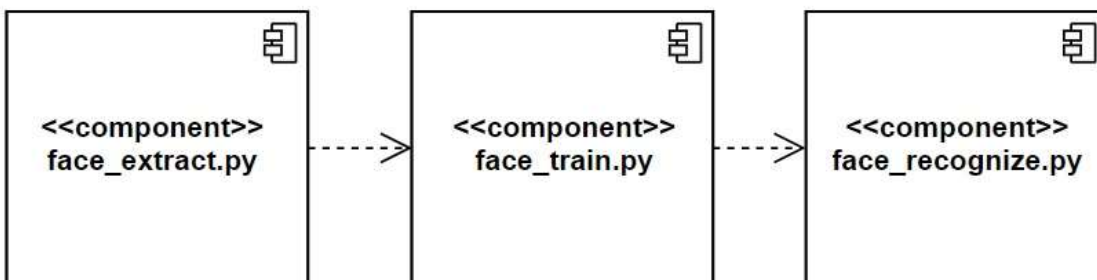
El presente módulo permite el procesamiento de información de rostros, comprende las fases de: extracción de características, clasificación y predicción.

En este módulo tendrán lugar el uso de librerías para el procesamiento de imágenes, redes neuronales convolucionales pre-entrenadas, algoritmos de pérdida, de detección, de aprendizaje automático y algoritmos de clasificación.

El módulo está constituido por tres submódulos elementales para su operación expuestos en la Figura 16, cada submódulo representa un programa en específico que posteriormente se ejecuta de manera secuencial, como se observa en la Figura 25.

Figura 25

Programas del módulo de reconocimiento facial



Requerimiento tecnológico

Los componentes tecnológicos utilizados para este módulo, tanto de hardware como de software se describen en la Tabla 4. Cabe recalcar que las características detalladas son las mínimas recomendadas.

Tabla 4*Características de los materiales empleados en el reconocimiento facial*

Software / Hardware	Función	Características Mínimas
S.O Linux	Sistema operativo de la tarjeta	Ubuntu 18.04 LTS
Python	Lenguaje de programación	Versión 3.6.9
OpenCV	Librería para el tratamiento de imágenes	Versión 4.1.1
Dlib	Librería de software multiplataforma de uso general	Versión 19.7.1
Jetson-Nano	Procesamiento del programa	GPU-128-core-NVIDIA Maxwell, Quad-core ARM A57, 4 GB de 64 bits LPDDR4, 4K @ 60 fps (H.264/H.265)
Raspberry camera	Captura de imágenes en diferentes ángulos	Sony IMX219, resolución fija 640x480, 60 FPS, V4L2

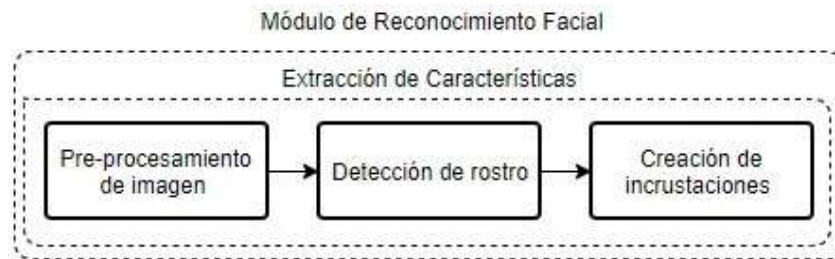
Extracción de características

El presente submódulo representa la obtención de las características del rostro mediante incrustaciones faciales. Una incrustación facial es una representación numérica de vectores que corresponden a cada cara detectada en un espacio de 128 dimensiones.

Para conseguir lo anterior mencionado se debe cumplir con las etapas que definen este submódulo como se muestra en la Figura 26.

Figura 26

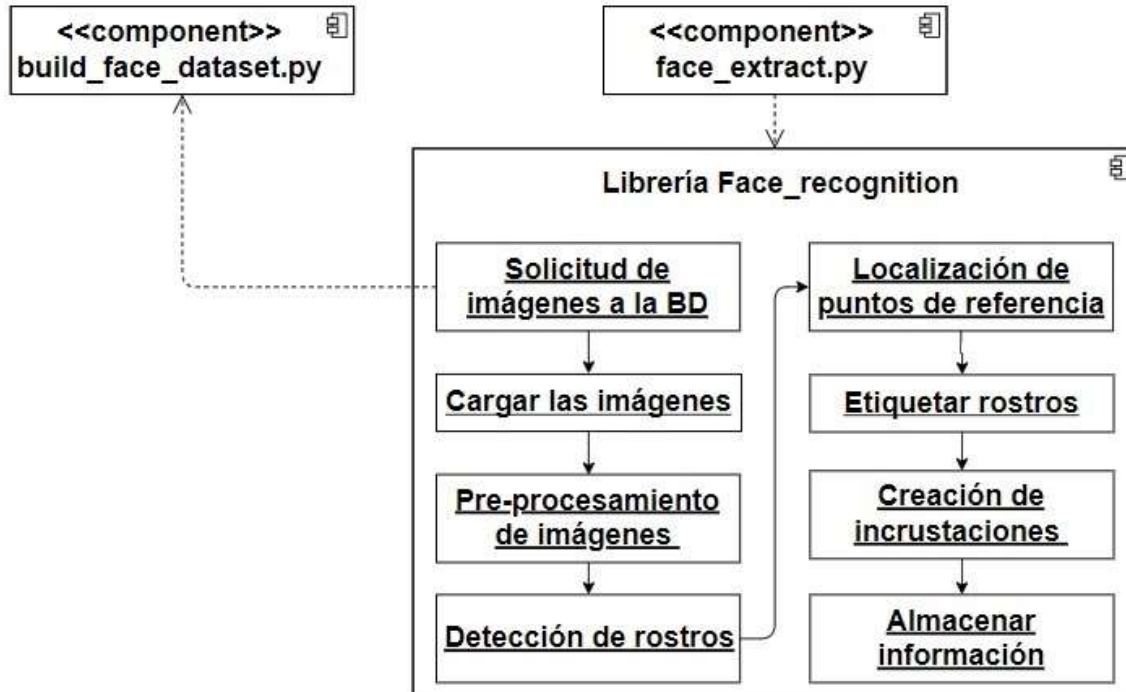
Etapas del submódulo de extracción de características



Las etapas del submódulo de extracción de características están contenidas en el programa "face_extract.py", el cual obedece los pasos representados en el diagrama de la Figura 27.

Figura 27

Diagrama de componentes del programa face_extract.py



Las librerías que se detallan en la Tabla 5 son las que se emplean en el programa para lograr la extracción de características faciales.

Tabla 5

Librerías empleadas en el módulo de extracción de características

Librería	Función
OpenCV	Proporciona la red pre-entrenada de aprendizaje profundo para el tratamiento de imágenes
Face_recognition	Permite la detección de rostros mediante la red pre-entrenada de Dlib y la extracción de características faciales
Numpy	Proporciona un objeto de matriz multidimensional de alto rendimiento y herramientas para trabajar con estas matrices
PIL	Carga un archivo de imagen (jpg, png, jpeg) y devuelve la imagen contenida en un vector
Os	Permite la manipulación de directorios ayudando con la búsqueda de las imágenes en la base de datos

Preprocesamiento de imagen

En esta etapa se procede a cargar las imágenes (obtenidas en el módulo de registro) dentro de una matriz *numpy* respectivamente, esto se realiza mediante la función *load_image_file* de la librería *face_recognition*.

A continuación, se emplea la técnica BLOB (Binary Large Object) para el tratamiento de la imagen. La técnica consiste en extraer una región de la imagen en la que algunas propiedades como la intensidad o el color son similares (Valenza, Nardelli, & Lanatá, 2013)

Mediante un módulo de red neuronal profunda (dnn) proporcionada por la librería *OpenCV* se consigue la aplicación de la técnica de una manera simple a través de la siguiente línea de código.

```
cv2.dnn.blobFromImage
```

El empleo de este comando requiere 5 parámetros de especificación: la imagen, el factor de escalamiento, el tamaño de la imagen, los valores de la media y el tipo de canal.

```
blob=cv2.dnn.blobFromImage(image, scalefact=1.0, size, mean, swapRB=True
```

El uso de “cv2.dnn.BlobfromImage” permite la transformación de la imagen a un formato estándar para su posterior ingreso en una red neuronal pre-entrenada y realiza acciones que se detallan a continuación (Rosebrock, 2018).

- Resta media: realiza una operación de sustracción entre los valores preestablecidos de la dnn ($R_{mean}, G_{mean}, B_{mean}$) y la cantidad promedio de RGB de una imagen (μ_R, μ_G, μ_B) que está determinada por la cnn de detección; en términos matemáticos es lo siguiente.

$$R = R_{mean} - \mu_R$$

$$G = R_{mean} - \mu_G$$

$$B = R_{mean} - \mu_B$$

- Factor de escala: normaliza los valores de RGB dividiéndolos para un factor sigma, este factor puede ser configurado manualmente a 1 (valor adoptado).

$$R = \frac{(R_{mean} - \mu_R)}{\sigma}$$

$$G = \frac{(R_{mean} - \mu_G)}{\sigma}$$

$$B = \frac{(R_{mean} - \mu_B)}{\sigma}$$

- Tamaño: redimensiona el tamaño de la imagen que la red neuronal convolucional de la siguiente etapa espera, en este caso, esto es 640x480.

El Blob creado con línea de código a continuación, usa los parámetros de la Tabla 6., para la etapa de detección de rostros.

```
imageBlob = cv2.dnn.blobFromImage(cv2.resize(image, (640,480)),1,
(640,480), (104,177,123), swapRB=False, crop=False)
```

Detección de rostro

Para este propósito se emplea la red neuronal convolucional (CNN) provista por Dlib denominada “mmod_human_face_detector.dat.bz2”, un modelo pre-entrenado, basado en el método MMDO (Max-Margin Object Detection).

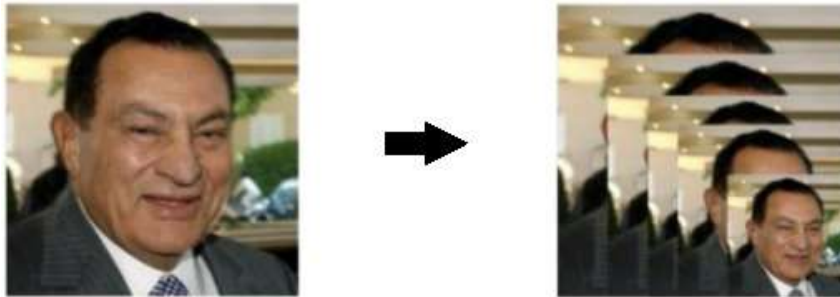
El motivo para usar modelos pre-entrenados es debido a que tanto los tiempos de respuesta como el costo computacional son menores (Rosebrock, 2018). La CNN identifica caras en la imagen de entrada y devuelve la posición de cada cara en la imagen a través de las coordenadas de un rectángulo que delimita la cara encontrada.

Para este fin la CNN cumple con el siguiente procedimiento:

- Toma la imagen y la pasa por 3 capas de disminución de muestras. Estas capas reducen el tamaño de la imagen en 8 pixeles (pirámide de imagen) y generan un mapa de características de 32 dimensiones, es decir, detectan características de bajo nivel como colores y bordes. En la Figura 28 se puede apreciar un ejemplo (Rajasegaran, Jayasundara, & Jayasekara, 2019)

Figura 28

Imagen luego de las capas de reducción

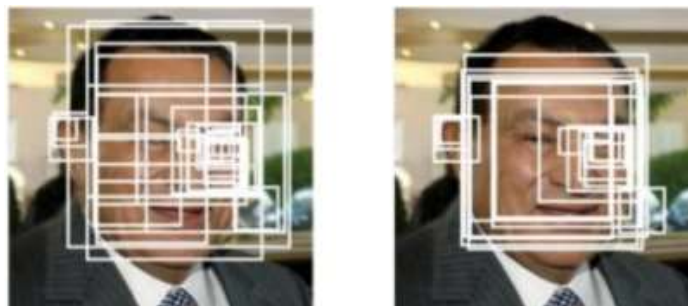


Nota. Adaptado de “*Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks*” (p.1502), por K. Zhang, Y. Qiao, 2016, IEEE Signal Processing Letters.

- Luego el resultado pasa por 4 capas convolucionales en donde se emplea ventanas deslizantes que recorren todos los valores del mapa obtenido, seleccionando regiones de interés y encerrando estas en cuadros delimitadores, como se observa en la Figura 29.

Figura 29

Localización de las regiones de interés (ROI)



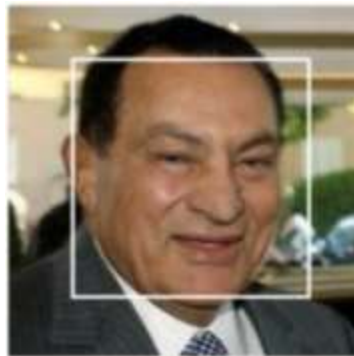
Nota. Adaptado de “*Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks*” (p.1502), por K. Zhang, Y. Qiao, 2016, IEEE Signal Processing Letters.

La selección del ROI se aplica mediante un puntaje de confiabilidad en cada cuadro delimitador; un puntaje de confiabilidad alto significa que una región de interés se encuentra en el cuadro delimitado.

- Finalmente, la última capa es la capa de pérdida, esta calcula un valor que indica qué tan bien está funcionando la CNN en la tarea de detección, es decir, realiza la clasificación de los cuadros delimitadores en función de las puntuaciones a través del algoritmo MMDO. La Figura 30 muestra el resultado final de una imagen al pasar por la CNN (King, 2016)

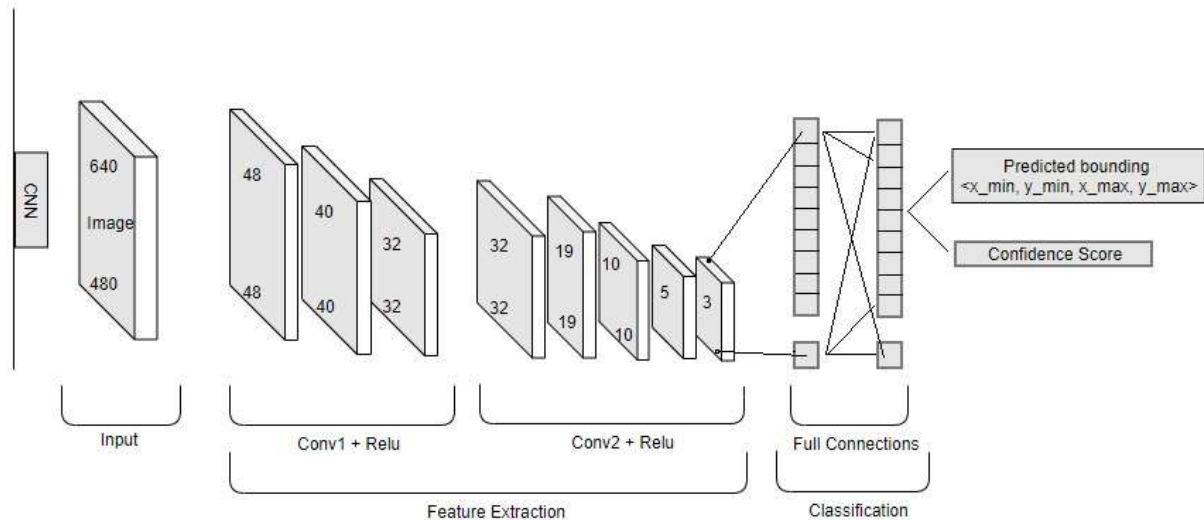
Figura 30

Detección de rostro mediante CNN



Nota. Adaptado de “*Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks*” (p.1502), por K. Zhang, Y. Qiao, 2016, IEEE Signal Processing Letters.

En la Figura 31 se puede observar cómo está estructurada la Red Neuronal Convolutiva para la etapa de detección.

Figura 31*Estructura de la CNN empleada*

La función Relu es usada en la red neuronal como función de activación con una tasa de aprendizaje de $lr=0.01$ y se emplea la técnica de normalización por lotes para normalizar la capa de entrada, ajustando y escalando las activaciones; consiguiendo mejorar la velocidad, el rendimiento y la estabilidad de la red. (Sridhar, 2018)

La presente CNN fue capacitada en un grupo de 6975 imágenes etiquetadas manualmente, fue creada a partir de varios conjuntos de datos (ImageNet, AFLW, Pascal VOC, VGG y WIDER), logrando una tasa de detección del 89.06%. (King, 2016)

Cabe recalcar que el uso de esta herramienta requiere que los valores de la imagen de entrada estén estandarizados, estos se detallan en la Tabla 6.

Tabla 6*Parámetros del modelo empleado*

Modelo	Escala	Tamaño píxeles (largo x ancho)	Resta Media	Orden de canales
mmod_human_face_detector.dat.bz2	1	640x480	104,177,123	RGB

Procedimiento para detectar rostros

El primer paso es conseguir el modelo pre-entrenado, este puede ser descargado desde: http://dlib.net/files/mmod_human_face_detector.dat.bz2 como un archivo .dat. Ahora se procede a crear una instancia de la clase *cnn_face_detection_model_v1* pasando los pesos pre-entrenados del archivo descargado.

```

""" Creación del objeto detector de cara con CNN pre-entrenada """
detector=dlib.cnn_face_detection_model_v1('mmod_human_face_detector.
dat')

```

A continuación, se prosigue con la detección de rostros haciendo uso del objeto creado, especificándole la imagen previamente procesada y el número de muestreo. El número 1 en el segundo argumento indica que se muestrea la imagen una vez. Esto hará que todo sea más grande y permitirá detectar más caras.

```

""" Uso del objeto para obtener detecciones """
face_locations = detector (face_image, number_of_times_to_upsample=1)

```

La variable *face_locations* es una lista de objetos que abarca objetos del tipo *rect* (representa un área rectangular en una imagen), los cuales contienen funciones *top*, *right*, *bottom*, *left* que devuelven las coordenadas correspondientes de todas las ubicaciones

de rostros en el marco. Para obtener dichas coordenadas, se recorre las ubicaciones identificadas y se dibuja un rectángulo alrededor del rostro.

```

""" Recorrer las ubicaciones de la cara"""
for face_location in face_locations:
    top = 4*face_location.rect.top()
    top = 4*face_location.rect.right()
    top = 4*face_location.rect.bottom()
    top = 4*face_location.rect.left()

""" Dibujar rectángulo alrededor de la cara"""
cv2.rectangle(face_image, (left, top), (right, bottom), (0,0,255),2)

```

Una vez que se hayan dibujado los rectángulos en los rostros identificados, lo que sigue es presentar los resultados. Esto se realiza mediante la función `imshow()` propia de OpenCv.

```

""" Visualizar imagen resultante"""
cv2.imshow('Dlib_CNN', face_image)

```

Creación de incrustaciones

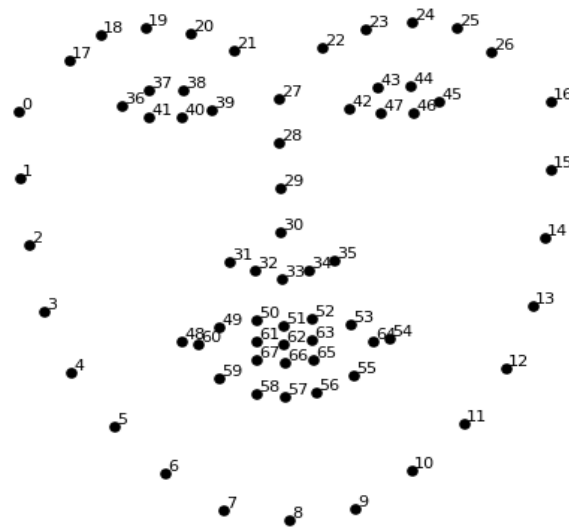
Esta etapa analiza los rostros provenientes de la etapa de detección y genera un vector de codificación facial de 128D (incrustación facial) para cada rostro detectado. Cada uno de los 128 valores representan un componente ortogonal de la codificación de la cara, es decir, dos imágenes diferentes de la misma persona tendrían una codificación similar, mientras que imágenes de dos personas diferentes tendrían una codificación totalmente diferente.

Para este propósito inicialmente es necesario estimar la ubicación de los puntos de referencia en el rostro. Para ello se emplea el predictor de forma de 68 puntos de referencia incorporado en la librería `Face_recognition`. El predictor fue pre-entrenado en

el conjunto de datos iBug 300-W permitiendo determinar puntos específicos en las siguientes regiones faciales: boca, ceja derecha-izquierda, ojo derecho-izquierdo, nariz y mandíbula distribuidas en la Tabla 7. Los índices de los 68 puntos se pueden visualizar en la Figura 32.

Figura 32

68 puntos de referencia en un rostro



Nota. Adaptado de “Speaker Inconsistency Detection in Tampered Video” (p. 5), 2018, European Signal Processing Conference.

Tabla 7

Distribución de los 68 puntos de referencia en un rostro

Sección del rostro	Distribución
Mandíbula	0-16
Ceja derecha	17-21
Ceja izquierda	22-26
Nariz	27-35
Ojo derecho	36-41
Ojo izquierdo	42-47
Boca	48-68

Dicho predictor es una implementación de *One Millisecond Face Alignment with an Ensemble of Regression Trees* presentado en 2014 por Kazemi y Sullivan. Esta técnica utiliza una función simple y rápida (diferencias de intensidades de píxeles) para estimar directamente las posiciones de referencia. Estas posiciones estimadas se refinan posteriormente con un proceso iterativo realizado por una cascada de regresores. Los regresores producen una nueva estimación de la anterior, tratando de reducir el error de alineación de los puntos estimados en cada iteración.

La Figura 33 muestra el resultado de la aplicación del predictor de forma de 68 puntos de referencia en una imagen de la base de datos.

Figura 33

Mapeo de 68 puntos faciales en una imagen de la base de datos



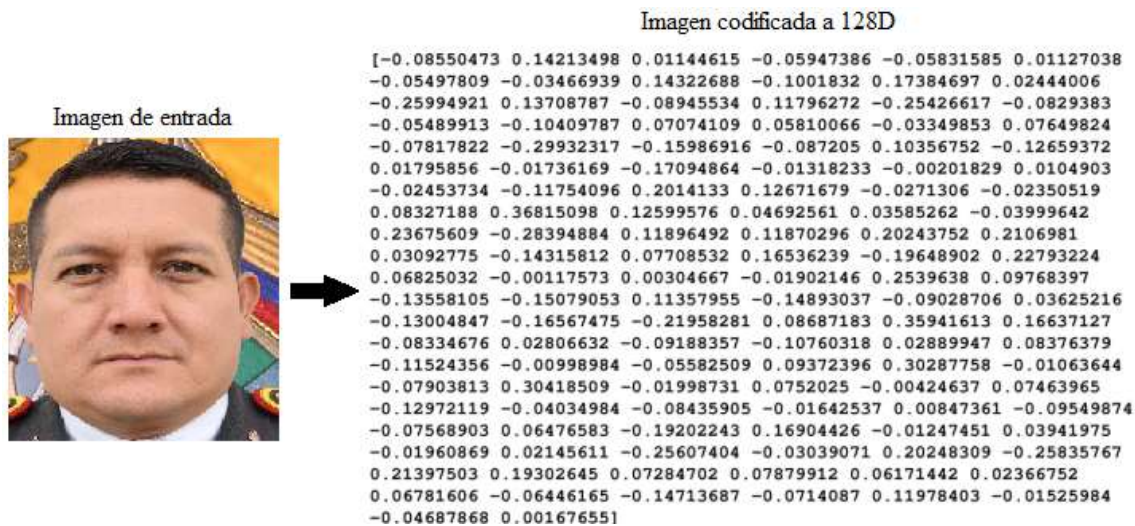
Una vez localizados los puntos de referencia solo queda obtener la codificación facial de 128 valores, para tal motivo se utiliza la CNN provista por Dlib denominada, *dlib_face_recognition_resnet_model_v1.dat.bz2*. Este modelo es una red ResNet (Deep Residual Learning for Image Recognition) con 29 capas de convolución, esencialmente es una versión de la red ResNet-34 del documento Deep Residual Learning for Image Recognition de He, Zhang, Ren y Sun con algunas capas eliminadas y la cantidad de filtros por capa reducida a la mitad (King, 2016).

Es una CNN pre-entrenada en un conjunto de datos de 3 millones de caras aproximadamente, conformado en su mayoría por el conjunto de imágenes de VGG y face Scrub.

La red está entrenada de una manera que genera un descriptor de 128 dimensiones (128D), utilizado para cuantificar un rostro, en la Figura 34 se presenta un ejemplo de pasar una imagen por la red.

Figura 34

Aplicación del descriptor 128D a una imagen de la BD



El proceso de entrenamiento de la red se comprende de datos de entrada y de la función de pérdida de triplete; cada lote de datos de entrada incluye un triplete de imágenes: la ancla, la imagen positiva y la imagen negativa. El ancla y la imagen positiva corresponden al rostro de una misma persona, mientras que la imagen negativa concierne al rostro de una persona diferente.

La red genera un descriptor 128D para cada una de las imágenes, modificando ligeramente los pesos de la red para acercar los vectores que corresponden a la misma persona y alejar los vectores que no corresponden a la misma. La modificación de estos

pesos se la realiza a través de un valor (0,4) de distancia guía, establecido por la propia red para agrupar vectores similares. Este valor de distancia (0,4) será posteriormente tomado como referencia para la clasificación entre clases.

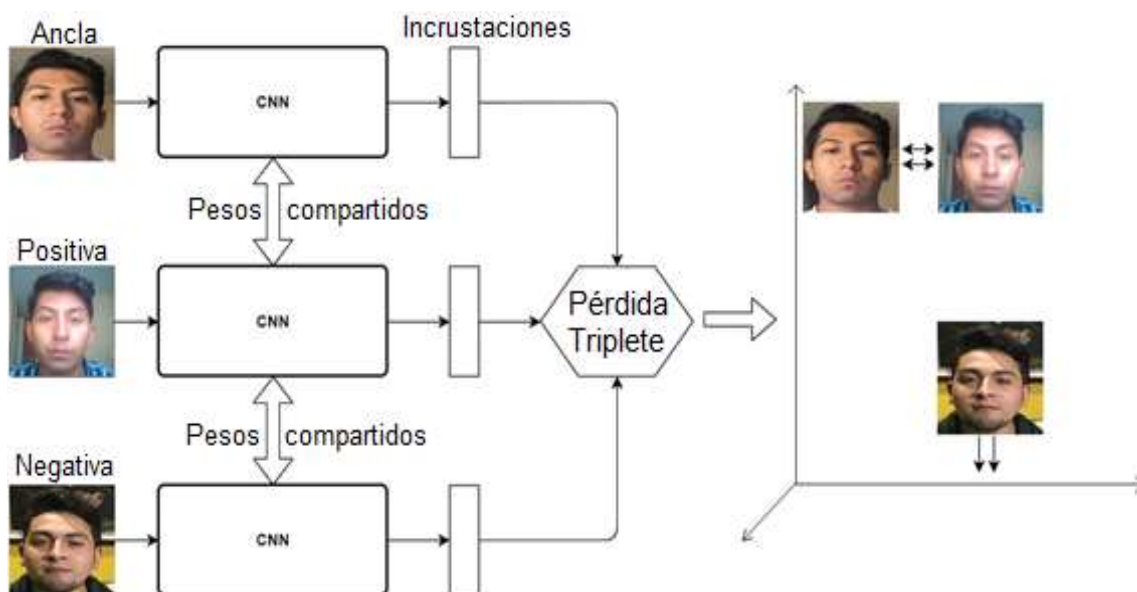
La función de pérdida de triplete formaliza esto acercando los descriptores 128D de dos imágenes de la misma persona, mientras que aleja los descriptores 128D de dos imágenes de diferentes personas (Fernández, 2019).

Esto quiere decir que, si a dos vectores descriptores de caras se le aplica una métrica de distancia (distancia euclidiana) y entre ellos existe una distancia inferior a 0,4, se puede considerar que pertenecen a la misma persona. De lo contrario, son de diferentes personas.

De esta manera, la red aprende a codificar rostros y devolver incrustaciones altamente robustas y discriminatorias, adecuadas para el reconocimiento de rostros, en la Figura 35 se aprecia un ejemplo del proceso.

Figura 35

Proceso de entrenamiento de la Resnet-29



Procedimiento para crear incrustaciones

El primer paso es conseguir todos los modelos necesarios descritos en la Tabla 8, estos pueden ser obtenidos a través de la página oficial de Dlib.

Tabla 8

Modelos pre-entrenados

Modelo	Descripción
mmod_human_face_detector.dat.bz2	Permite la detección de rostros
shape_predictor_68_face_landmarks.dat	Permite la ubicación de puntos de referencia en el rostro
dlib_face_recognition_resnet_model_v1.dat.bz2	Permite la generación de incrustaciones faciales

Ahora se procede a crear instancias de cada una de las clases de los respectivos modelos pre-entrenados.

```
detector=dlib.cnn_face_detection_model_v1('mmod_human_face_detector.dat')
pose_predictor_68_point=dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')
face_encoder=dlib.face_recognition_model_v1('dlib_face_recognition_resnet_model_v1.dat')
```

A continuación, se realiza la detección de rostros mediante el uso del objeto *detector*, para luego con el objeto *pose_predictor_68_point* lograr ubicar puntos de referencia en cada uno de los rostros detectados.

Finalmente, se prosigue con el cálculo de las incrustaciones para cada rostro que fue detectado y mapeado respectivamente. Para ello se emplea la función *face_encoder.compute_face_descriptor ()* de Dlib.

Todo lo anterior mencionado se encuentra definido dentro de la función *face_encodings* de la librería *Face_recognition*; esta devuelve un descriptor de 128D contenido en un vector Numpy para cada rostro en una imagen.

```
def face_encodings (face_image,number_time_upsample=1,num_jitters=1):
    """ Detección de rostro"""
    face_locations = detector (face_image, number_time_upsample=1)
    """ Detección de puntos de referencia en el rostro"""
    raw_landmarks = [pose_predictor_68_point(face_image,
    face_location) for face_location in face_locations]
    """ Cálculo de incrustaciones para cada rostro"""
    return[np.array(face_encoder.compute_face_descriptor(face_image
    ,raw_landmark_set,num_jitters)) for landmark_set in
    raw_landmarks]
```

Cabe recalcar que para cada rostro codificado hubo un paso previo de etiquetado con su respectivo nombre, consiguiendo así almacenar en dos vectores la información del nombre (*names= []*) y la incrustación del rostro (*embeddings= []*) respectivamente.

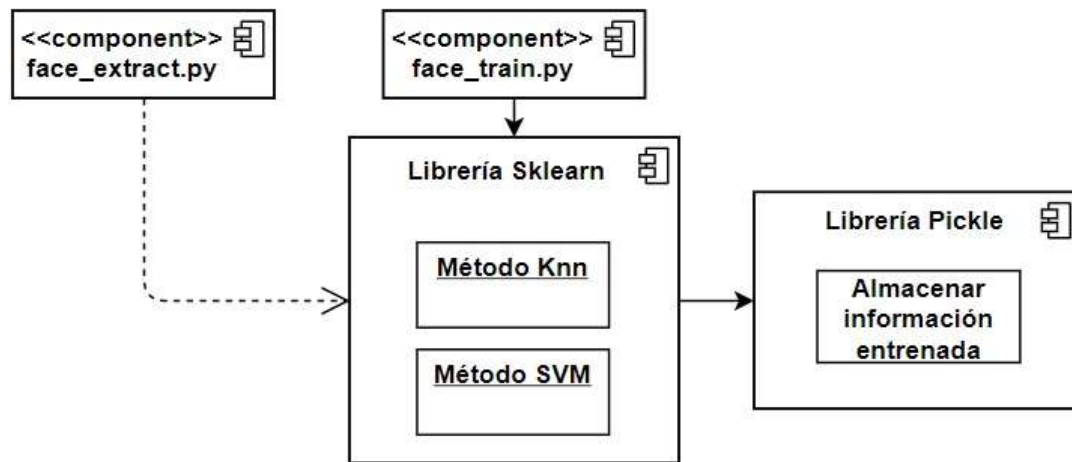
Entrenamiento

Una vez que los rostros hayan sido codificados y etiquetados respectivamente, el siguiente paso es entrenar estos datos a través de un método de aprendizaje automático.

En el presente proyecto se hace uso de 2 métodos de aprendizaje automático, para su posterior comparación. Para ello se ejecuta el programa “*face_train.py*” el cual cumple con los pasos de la Figura 36.

Figura 36

Diagrama de componentes del programa *face_train.py*



Las librerías que se detallan en la Tabla 9 son las que se emplean en el programa para conseguir el entrenamiento de la información.

Tabla 9

Librerías de entrenamiento

Librería	Función
Sklearn	Biblioteca de aprendizaje automático gratuita para Python que facilita la importación de los métodos de aprendizaje Knn y SVM
Pickle	Permite el almacenamiento de la información procesada implementando protocolos binarios.

Cabe recalcar que el conjunto de datos fue de 337 imágenes de rostros (obtenidas de 10 personas diferentes).

Entrenamiento con KNN

En primer lugar, se emplea el algoritmo *Knn* (*K nearest neighbors*) que clasifica los datos empleando medidas de distancia (distancia euclidiana) y una ponderación (voto por etiquetas).

El primer paso para emplear el algoritmo *Knn* es importar la librería de aprendizaje automático.

```
from sklearn import neighbors
```

A continuación, se procede a definir el valor de K , el cual representa el número de vecinos más cercanos e influye de gran manera en la precisión de la predicción.

Para ello se calcula la raíz cuadrada del número total de muestras, es decir, el total de rostros codificados, que en este caso será 100.

```
n_neighbors = int(round(math.sqrt(len(X))))
```

Luego se procede a la creación del método especificándole el valor de K , el algoritmo utilizado para calcular los vecinos más cercanos y la función de peso utilizada en la clasificación.

```
knn_clf=neighborsClassifier(n_neighbors=n_neighbors,  
algorithm='ball_tree',weights='distance')
```

La función de peso empleada es la función de distancia inversa, su objetivo es dar más peso a los puntos que están cerca y menos peso a los puntos que están más lejos.

Una vez creado el método se pasa a entrenar los datos de etiquetas e incrustaciones mediante la instanciación del método con la función *fit* ().

```
knn_clf.fit(X,Y)
```

Finalmente, se almacena la información entrenada a través de la creación de un objeto de la librería Pickle.

```
with open (model_save_path, 'wb' as f:  
    pickle.dump(knn_clf, f))
```

Entrenamiento con SVM

El segundo método a emplear es el algoritmo *SVM* (*support vector machine*), tiene la misma tarea de clasificación de datos, con la diferencia que para hacerlo usa hiperplanos de decisión, márgenes y vectores de soporte. Cabe recalcar que para poder emplear este método se debió codificar las etiquetas (nombres) de los rostros; esto se lo efectuó mediante una función propia de la librería Sklearn.

La implementación de este algoritmo se desarrolló en el mismo programa "face_train".py, haciendo uso de las mismas librerías de la Tabla 6.

El primer paso para emplear el algoritmo *SVM* es importar la librería de aprendizaje automático y la función de codificación de etiquetas.

```
from sklearn.svm import SVC  
from sklearn.preprocessing import LabelEncoder
```

Seguidamente se procede con la codificación de las etiquetas empleando la función importada dotándole de los datos almacenados en el vector "names".

```
le = LabelEncoder ()
labels = le.fit_transform(names)
```

A continuación, se lleva a cabo el proceso de entrenamiento mediante la llamada al método, proporcionándole los parámetros de ajuste y los datos de etiquetas e incrustaciones a entrenar.

```
train_svm = SVC(C=1.0, kernel="linear", probability=True)
train_svm.fit(embeddings, labels)
```

Donde “C” es un parámetro de rigor y controla cuánto desea evitar clasificar erróneamente cada punto de datos en el conjunto de entrenamiento. El segundo parámetro es el tipo de *kernel* que va a usar el algoritmo en la construcción del límite de decisión y el parámetro final corresponde al parámetro que produce estimaciones de probabilidad a partir de valores de decisión. (Rosebrock, 2018)

Por último, se almacena la información entrenada mediante la creación de un objeto de la librería *Pickle*.

```
With open(model_save_path, 'wb') as f:
    pickle.dump(train_svm, f)
```

Predicción

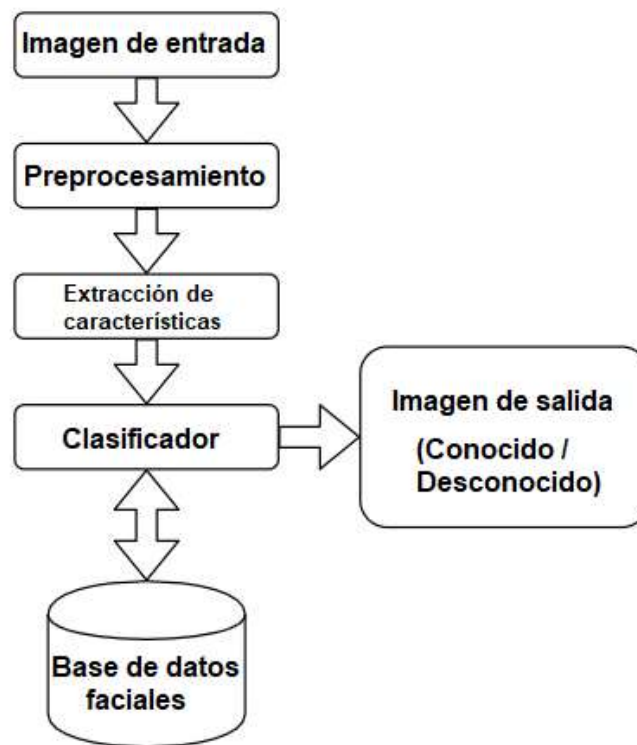
El presente submódulo tiene por objetivo recibir la información de rostros (instancias de datos), procesar dicha información y de acuerdo al clasificador determinar su identificación (clase).

Es decir, esta fase es la encargada de predecir si el rostro es conocido o no, en base a resultados previamente proporcionados. Si el rostro es conocido, lo identifica con

su nombre correspondiente, caso contrario lo identifica como desconocido. La Figura 37 muestra la manera en la que el proceso de predicción se lleva a cabo.

Figura 37

Diagrama de proceso de predicción



Lo anterior mencionado está contenido en el programa “face_recognize.py” en el cual se hace uso de las librerías que se detallan en la Tabla 10.

Tabla 10

Librerías de predicción

Librería	Función
OpenCV	Permite el preprocesamiento y tratamiento de las imágenes.
Face_recognition	Permite la detección de rostros y la extracción de características faciales
Sklearn	Permite el empleo de los métodos de aprendizaje para la clasificación de rostros.

Predicción con KNN

La tarea del método Knn es realizar la predicción de una persona en imágenes al encontrar las K caras más similares (imágenes con rasgos faciales cercanos a una distancia euclidiana) en su conjunto de entrenamiento y realizar un voto mayoritario (ponderado) en su etiqueta. (Geitgey, 2018)

En este proyecto se considera la existencia de una coincidencia facial cuando la distancia euclidiana entre la nueva imagen ingresada y las imágenes del conjunto de entrenamiento es menor a 0,4 (valor establecido como umbral). Debido a que con este valor se obtiene menos errores de predicción y es el valor que se recomienda emplear en correspondencia con la CNN utilizada en la etapa de creación de incrustaciones (King, 2016).

Por ejemplo, si $K = 3$, y las tres imágenes de la cara más cercanas (distancia euclidiana $< 0,4$) a la imagen dada en el conjunto de entrenamiento son 1 imagen de David y 2 imágenes de William, el resultado sería "William".

Para lo antes mencionado se emplean predicciones de clase, mediante la función de predicción `predict()`, propia de la librería Sklearn.

```
pred = knn_clf.predict(faces_encodings)
```

Cabe recalcar que el método no devuelve una medida de probabilidad de acierto, solamente devuelve la etiqueta (nombre de la persona), de acuerdo la distancia entre caras; por lo tanto, se convierte este puntaje de distancia facial en un puntaje porcentual de acierto mediante la función `face_distance_to_conf()`.

```
def face_distance_to_conf (face_distance, face_match_threshold=0.4):
```

A dicha función se le proporciona las predicciones de las distancias más cercanas de entre todos los rostros de la BD (el número de predicciones dependerá del tiempo de ejecución del sistema) y el umbral de distancia preestablecido.

Esto quiere decir que para distancias inferiores al límite de decisión (0,4) el puntaje porcentual de coincidencia va a ser mayor, este límite en términos de porcentaje implica una correspondencia del 70%.

Un ejemplo gráfico de la predicción a través del método Knn puede apreciarse en la Figura 38, en donde se observa que para la nueva instancia ingresada (william_3.jpg), con $K=3$ el clasificador escoge 3 imágenes del conjunto de entrenamiento que más cercanas estén a la incrustación de la nueva instancia.

Figura 38

Predicción de rostro con el método KNN



Los dos primeros valores (0.3913191431, 0.3998326483) cumplen con la característica de cercanía y con el umbral establecido (<0.4). Eso significa que las 2 imágenes (william_1.jpg, william_2.jpg) pueden considerarse la misma persona que la instancia ingresada. El tercer valor obtenido (0.9053700273) significa que la tercera imagen no es la misma persona que la nueva instancia. Por lo tanto, se realiza un voto mayoritario en la etiqueta "William" y esta se muestra como resultado con una predicción mayor al 70%.

Predicción con SVM

Por otra parte, este método realiza la tarea de predicción de una persona en imágenes mediante predicciones de probabilidad, es decir, dada una nueva instancia de datos, el método SVM devuelve la probabilidad de que dicha instancia corresponda a algún rostro con los que fue entrenado.

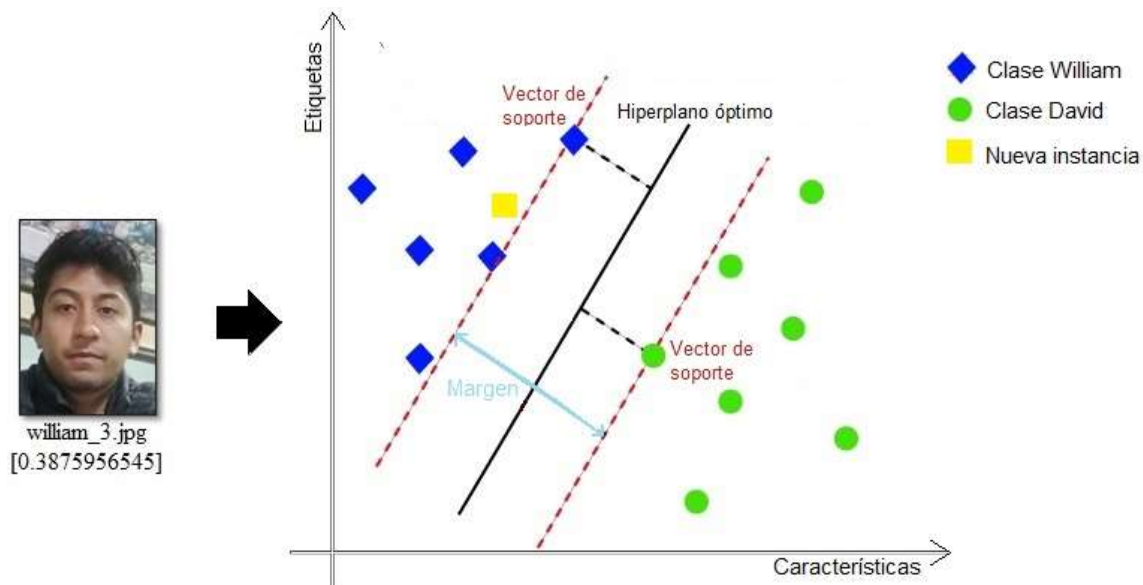
El método consigue este puntaje de probabilidad mediante la mayor cantidad de veces que se repita una etiqueta en el proceso de predicción.

Un valor del 99% significa una alta probabilidad de que la persona pueda pertenecer a un rostro del conjunto de entrenamiento, por tanto, sea correctamente identificada, por el contrario, un valor bajo del 40% significa una identificación errónea. (Obando, 2019)

Para lo antes mencionado se emplea la función *predict_proba* (), propia de la librería Sklearn.

```
preds = SVC.predict_proba(vec_encodings)[0]
```

Un ejemplo gráfico de la clasificación a través del método SVM se presenta en la Figura 39, en donde se observa que para la nueva instancia ingresada (william_3.jpg), el clasificador ubica la instancia entre las posibles clases y separa los puntos de datos utilizando un hiperplano con la mayor cantidad de margen.

Figura 39*Predicción con el método SVM*

Posteriormente, la salida del SVM se pasa como argumento de la mencionada función, ésta devuelve un vector con las probabilidades de que la instancia ingresada pertenezca a algún rostro (clase) del conjunto entrenado. Se escoge la clase con el mayor porcentaje de probabilidad (etiqueta que más veces se repitió), que para este caso es el décimo índice del vector, correspondiente a la etiqueta "William_Ibarra", como se observa en la Figura 40.

Figura 40*Predicción de una persona con el método SVM*

```
[ 0.00724584  0.01957415  0.00301479  0.00088014  0.00213936  0.0048686
 0.00163113  0.00223248  0.00247275  0.15952861  0.79641216]
10
0.796412156296
William_Ibarra
```

En el siguiente capítulo se comparará el rendimiento de los métodos Knn y SVM antes explicados.

Módulo de acceso

El presente módulo representa la incorporación del sistema de identificación y verificación facial a la chapa magnética, esto sumado a la elaboración de un historial de ingreso, el cual se ira actualizando en la base de datos de MySQL.

Incorporación sistema – chapa

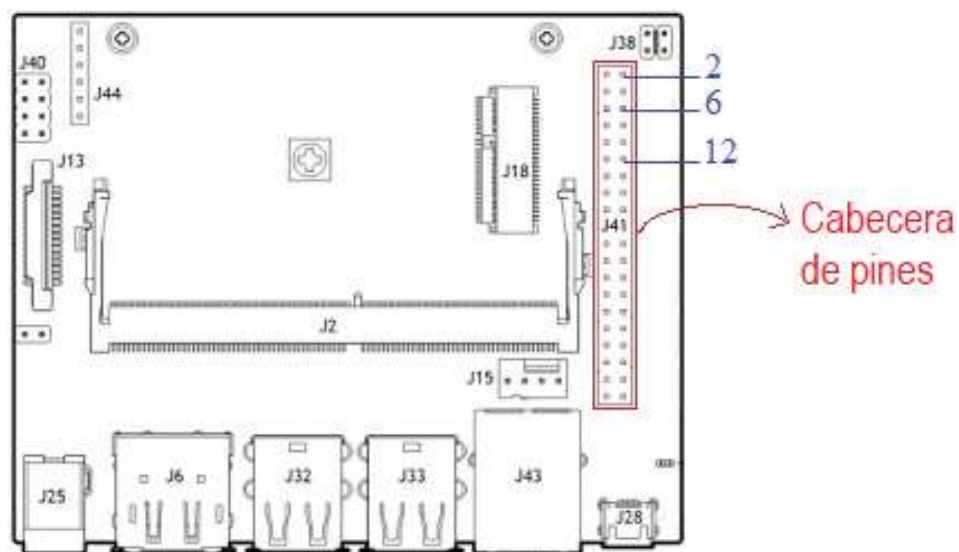
Una vez que el sistema haya reconocido el rostro de una persona, la tarjeta de desarrollo envía una señal de activación a la chapa magnética para que permita el ingreso de la persona reconocida.

La *Jetson Nano* incluye una cabecera de expansión con 40 pines GPIO (General Purpose Input Output). Por defecto todos los pines vienen configurados como GPIO, excepto los pines de alimentación, comunicación y sincronismo.

Para el control de la chapa magnética se emplea 3 pines en total: uno de alimentación (2), uno de tierra (6) y uno para la señal de control (12), como se muestra en la Figura 41.

Figura 41

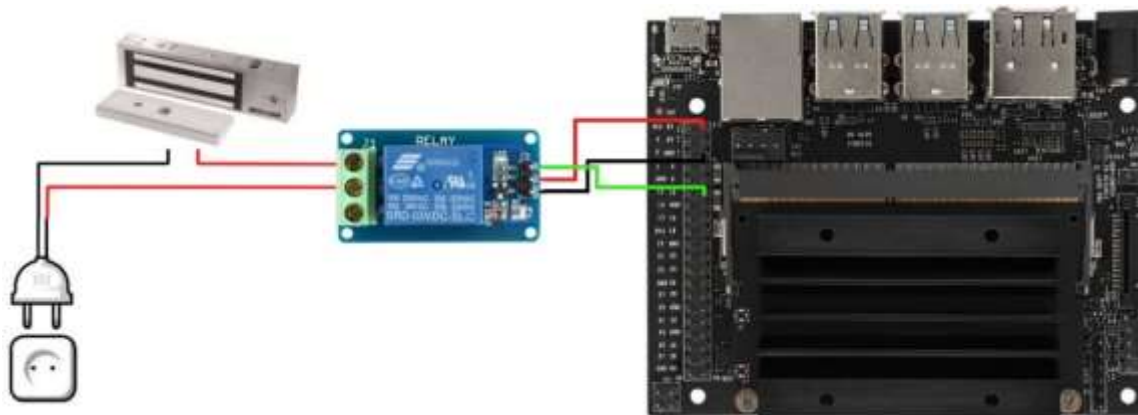
Pines de conexión de la chapa magnética



Debido a que el Nano solamente opera voltajes de 5V, este no puede controlar directamente a chapa magnética, la cual requiere de 12V para su funcionamiento. Por tanto, se empleó un módulo relé de 1 canal como dispositivo intermediario para la manipulación de diferentes voltajes, como se representa en la Figura 42.

Figura 42

Diagrama de conexión chapa-Jetson Nano

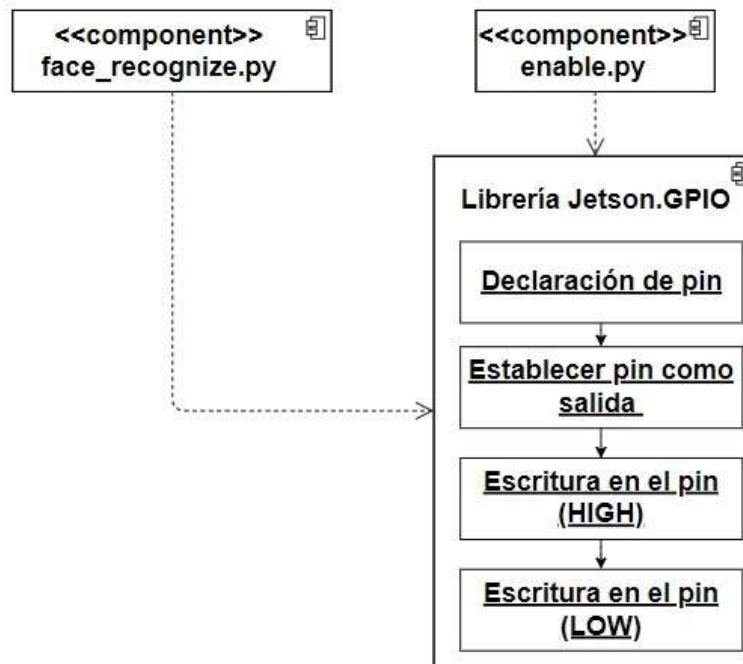


Una señal en alto en el pin 12 será la encargada de activar la chapa magnética, para ello se elabora el programa “enable.py”, el cual hace uso de la librería *Jetson.GPIO* (propia del sistema Nano) para la habilitación de pines.

Los procesos que se llevan a cabo en el programa se encuentran representados en la Figura 43.

Figura 43

Diagrama de los componentes del programa *enable.py*



Mediante la función *GPIO.output ()* se puede controlar el estado (High/Low) del pin, solamente se debe especificar el pin de salida y el estado deseado.

```
GPIO.output(output_pin=12, curr_value=GPIO.HIGH)
```

El tiempo de escritura en alto es de 2 segundos, lo que permite que el módulo relé se active y por ende se interrumpa el paso de corriente a la chapa, consiguiendo que la puerta se abra. Posteriormente se realiza la escritura en bajo para que el relé permita el paso de corriente a la chapa y el imán se magnetice, de tal manera que la puerta se bloquee hasta nueva orden.

Historial de ingreso

Una vez que la chapa magnética se haya activado, el ingreso de las personas autorizadas será registrado en una base de datos de MySQL, alojada en un servidor web local.

Para conseguir esto se requiere de la instalación de un *LAMP*, que es una plataforma de desarrollo web de código abierto que utiliza Linux como sistema operativo, Apache como servidor web, MySQL como sistema de gestión de bases de datos relacionales y Python como lenguaje de programación. Además, se instaló la herramienta de software gratuito *phpMyAdmin* para administrar MySQL a través de la web.

Las características técnicas del software empleado se detallan en la Tabla 11, cabe recalcar que las características detalladas son las mínimas recomendadas.

Tabla 11

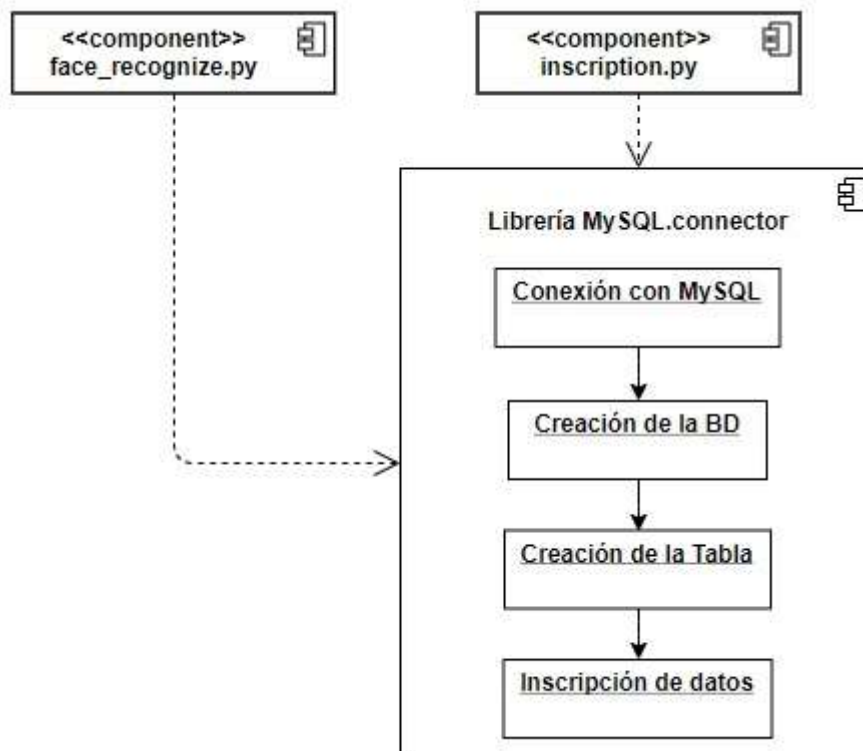
Software de instalación para el historial de ingreso

Software	Características
SO Linux	Distribución Ubuntu 18.04 LTS
Apache	Versión 2.4.43
MySQL	Versión 8.0.20
Python	Versión 3.6.9
phpMyAdmin	Versión 5.0.2

Después de que se instalaron las dependencias necesarias, se crea el programa “*inscription.py*”, el cual permite tomar las etiquetas de la fase de predicción e inscribirlas en una base de datos con información adicional de fecha y hora de ingreso. Los pasos que se cumplen en dicho programa se encuentran representados en la Figura 44.

Figura 44

Diagrama de los componentes del programa *inscription.py*



En el mencionado programa se hace uso de las librerías que se detallan en la Tabla 12.

Tabla 12

Librerías empleadas en la inscripción de usuarios

Librería	Función
MySQL.connector	Permite que los programas de Python accedan a las bases de datos MySQL, utilizando una API.
Datetime	Proporciona clases para manipular fechas y horas.

Procedimiento para la inscripción

El primer paso para inscribir las etiquetas (nombres) de las personas en la base de datos de MySQL desde Python, es importar las librerías necesarias.

```
import mysql.connector
from datetime import datetime
```

Seguidamente se crea el objeto que establece la conexión con el servidor MySQL, proporcionándole los parámetros que se detallan en la Tabla 13.

```
connection=mysql.connector.connect(host='localhost',user='root',password='root')
```

Tabla 13

Argumentos de conexión

Argumentos	Descripción
User(username*)	El nombre de usuario utilizado para autenticarse con el servidor MySQL.
Password(passwd*)	La contraseña para autenticar al usuario con el servidor MySQL.
Host	El nombre de host o la dirección IP del servidor MySQL.

A continuación, se crea la base de datos denominada “Registro”, en la cual se va a construir una tabla con la información de nombres, fecha y hora de ingreso.

```
cursor=connection.cursor()
cursor.execute(CREATE DATABASE Registro)
```

Posteriormente sobre la base de datos “Registro” se establece la tabla denominada “Personal”, con los campos detallados en la Tabla 14.

```
mysql_Create_Table_Query = """CREATE TABLE Personal (Id int (11))
AUTO_INCREMENT NOT NULL, Nombre varchar(250) NOT NULL, Fecha Date
NOT NULL, Hora Time NOT NULL, PRIMARY KEY (Id)"""
```

Tabla 14

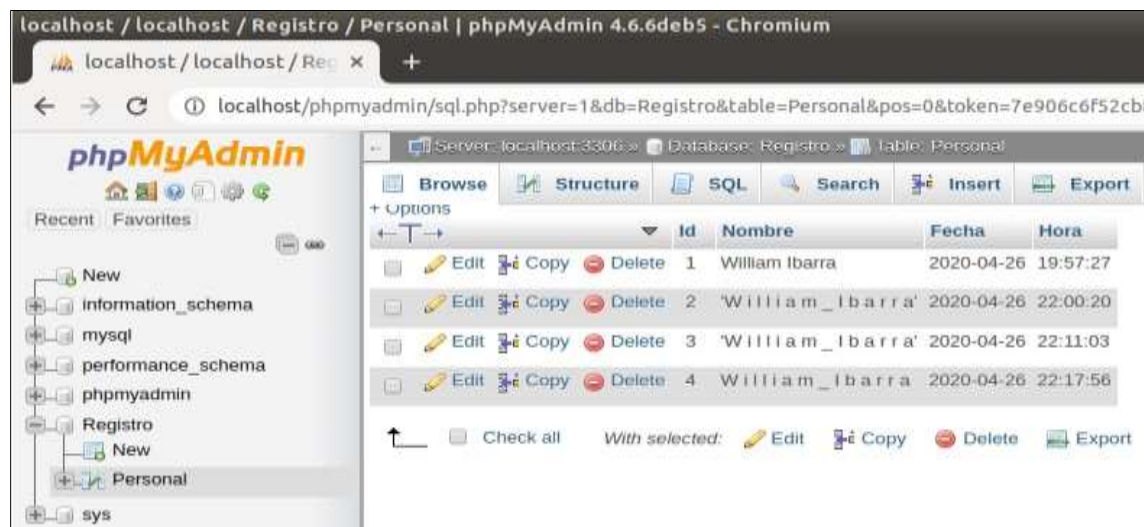
Campos de la tabla personal creada en MySQL

Argumentos	Descripción
Id	Campo numérico que genera una identidad única para nuevas filas, se incrementa automáticamente con cada nuevo registro
Nombre	Campo de caracteres proporcionado por la fase de predicción del módulo de reconocimiento facial.
Fecha	Campo Date que indica el año, mes y día en que se concedió el acceso, formato YY "/" mm "/" dd.
Hora	Campo Time que indica el momento en que se otorgó el acceso, formato H:M:S

El resultado final del proceso de inscripción de una etiqueta en la base de datos se puede observar mediante la herramienta phpMyAdmin en la Figura 45.

Figura 45

Inscripción del usuario William Ibarra en MySql



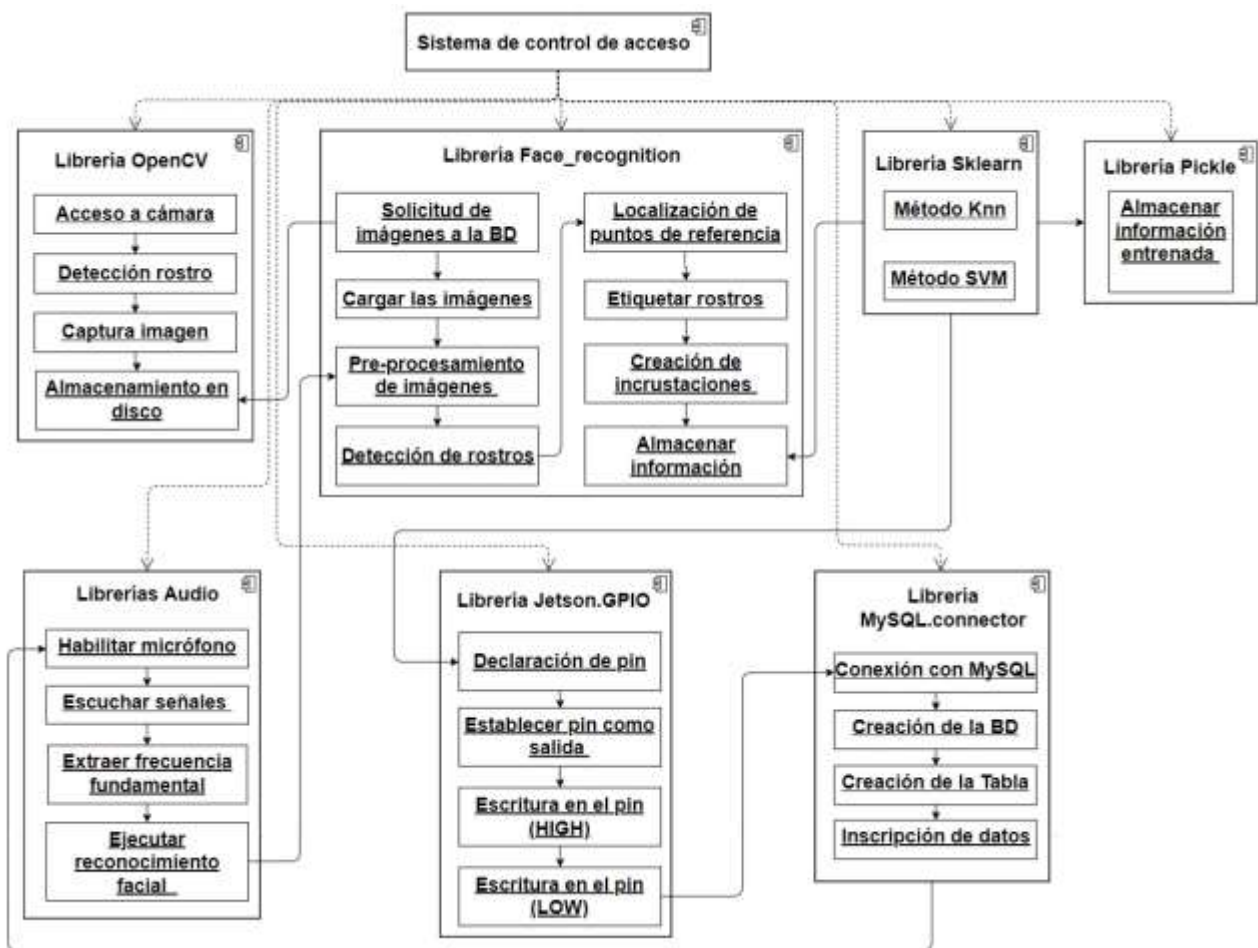
Integración

Descripción general del proceso

El sistema de control de acceso está constituido por los cuatro módulos explicados anteriormente, en la Figura 46 se presenta el diagrama de componentes del sistema en su totalidad.

Figura 46

Diagrama de componentes del sistema de control de acceso



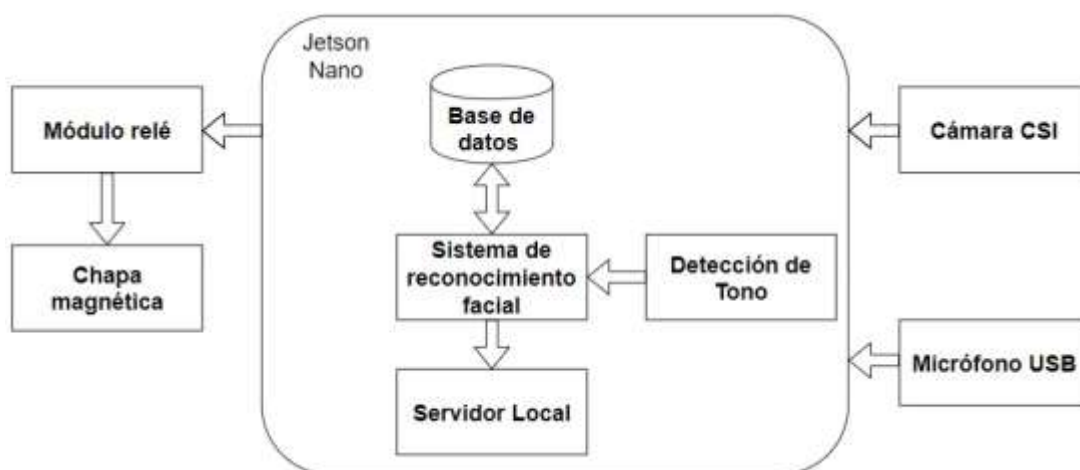
Para realizar el control del acceso, el sistema realiza los procedimientos que se detallan a continuación:

- Espera una señal de audio en el rango de frecuencias de 2000Hz a 2500Hz.
- Si la señal de audio es detectada, inmediatamente se activa la cámara para extraer información facial durante 5 segundos, en este mismo tiempo se realiza la comparación con imágenes de la base de datos y la toma de una decisión.
- Si existe un puntaje de coincidencia alto entre la persona que solicita el ingreso y la imagen de la base de datos se activa una salida de la tarjeta que permite la activación del relé y por ende el acceso.
- Al mismo tiempo que el sistema concede el acceso, este registra el nombre, fecha y hora de la persona que ingresó.
- Finalmente, el sistema vuelve al estado de escuchar, aguardando la señal de activación.

El diagrama de bloques presentado en la Figura 47 representa el flujo que adopta la información en el sistema de control de acceso.

Figura 47

Sistema de control de acceso



Por último, se presenta el sistema de control de acceso físicamente integrado en las Figuras 48 y 49.

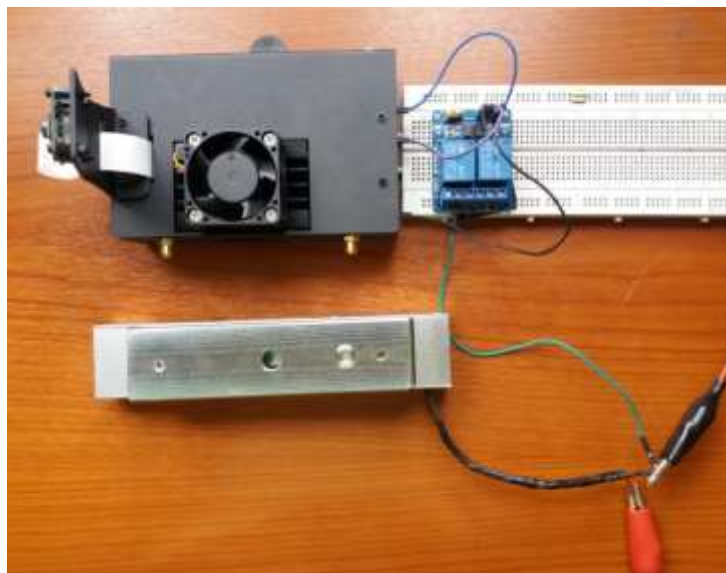
Figura 48

Vista frontal del sistema de control de acceso



Figura 49

Vista superior del sistema de control de acceso



Capítulo 4

Pruebas y resultados

El método de evaluación del rendimiento es el criterio para analizar la eficiencia de cualquier sistema de reconocimiento facial. La evaluación es esencial para comprender la calidad del modelo o la técnica, para refinar los parámetros en el proceso iterativo de aprendizaje y para seleccionar el modelo o la estrategia más adecuada de un conjunto dado de modelos o técnicas (WL-Chao,2011).

Este apartado representa la evaluación del rendimiento de cada método de aprendizaje en el proceso de identificación y verificación de rostros a través de dos pruebas básicas.

Inicialmente se da a conocer los tipos de pruebas a las cuales serán sometidos cada uno de los métodos de aprendizaje vistos con anterioridad y los parámetros en los cuales se fundamentan dichas pruebas.

Posteriormente se establece el contexto (espacio de pruebas y sujetos de prueba) en el que se van a desarrollar las pruebas.

Finalmente, se presenta un análisis comparativo, en el cual se muestra el método de aprendizaje que mejor rendimiento obtuvo realizando las tareas de identificación y verificación facial.

Pruebas

Las pruebas que van a desarrollarse se encuentran relacionadas con la variación de distancias y la variación de la luz natural en un ambiente no controlado, a fin de observar el rendimiento del sistema de identificación y verificación facial de acuerdo al método de aprendizaje empleado.

La primera prueba corresponde a la variable “distancia de reconocimiento del rostro del sujeto de prueba”, esto significa que se realiza la toma de datos de reconocimiento ubicando al sujeto de prueba a diferentes distancias.

La segunda prueba corresponde a la variable “iluminación del rostro del sujeto de prueba”, esto significa que se realiza la toma de datos de reconocimiento en horarios específicos donde la luz solar incide tanto en intensidad como en dirección sobre el rostro del sujeto de prueba.

Las Tablas 15 y 16, resumen las especificaciones de las pruebas a realizarse.

Tabla 15

Pruebas de distancia

Prueba	Detalle
Distancia 1	50 cm – luz normal
Distancia 2	80 cm – luz normal
Distancia 3	120 cm – luz normal

Los valores de las distancias fueron establecidos de acuerdo al sentido común y la lógica de las personas al momento de ubicarse frente la cámara para su reconocimiento.

Tabla 16

Pruebas de intensidad de luz

Prueba	Detalle
Iluminación 1	Cenital – 12 a 1 am
Iluminación 2	Poca luz – 5 a 6 pm
Iluminación 3	Contraluz – 8 a 10 am

Las condiciones de iluminación son establecidas de acuerdo a los intervalos de tiempo, los cuales determinan la dirección de la luz. Se mantiene la misma distancia para la medición.

Parámetros de evaluación

Para los intereses del presente proyecto se adoptarán las métricas detalladas en la Tabla 17.

Tabla 17

Métricas de evaluación

Métricas	Detalle
Precisión (%)	Medida probabilística que define qué tan bien una prueba de clasificación binaria identifica o excluye correctamente una instancia.
Tiempo de Clasificación (seg)	Intervalo de tiempo en el cual se demora cada método de aprendizaje en realizar las tareas de clasificación y mostrar sus resultados.

Contexto

El escenario en el cual se van a desarrollar las pruebas se encuentra en una zona semi-cubierta a fin de obtener resultados en un entorno real para el cual está diseñado el sistema de reconocimiento facial; este se presenta en la Figura 50.

Figura 50*Entorno de pruebas*

El conjunto de prueba está formado por 11 personas que se encuentran dentro de la base de datos y 1 que no lo está. Para el registro de la información fotográfica se siguió el procedimiento tratado en la etapa de adquisición de datos (capítulo 3) obteniendo los resultados de la Tabla 18.

Tabla 18*Sujetos de prueba*

Sujetos	Cantidad de fotografías
Usuario 1	30
Usuario 2	33
Usuario 3	43
Usuario 4	37
Usuario 5	32
Usuario 6	34
Usuario 7	32
Usuario 8	34
Usuario 9	31
Usuario 10	30
Desconocido	10

Desarrollo de pruebas

Una vez establecido el conjunto de prueba y las métricas a determinar, se procede a realizar la tabulación de los datos, los cuales se presentan en las tablas a continuación. Cabe recalcar que un puntaje de coincidencia es considerado como admisible cuando su valor es mayor al 80% y que se estableció un tiempo de 5 segundos en los cuales el sujeto de prueba debe mantenerse alineado a la cámara para la toma de datos.

Pruebas con KNN

El método de aprendizaje tanto para las pruebas de distancia como para las de iluminación se configuró con los parámetros de $K=3$ y con un umbral de decisión igual a 0,4; explicados en el capítulo 3.

Prueba 1 – distancia

En la Figura 51 se presenta un ejemplo tomado a Distancia 1, con el sujeto de prueba usuario 4.

Figura 51

Sujeto de prueba identificado a distancia 1

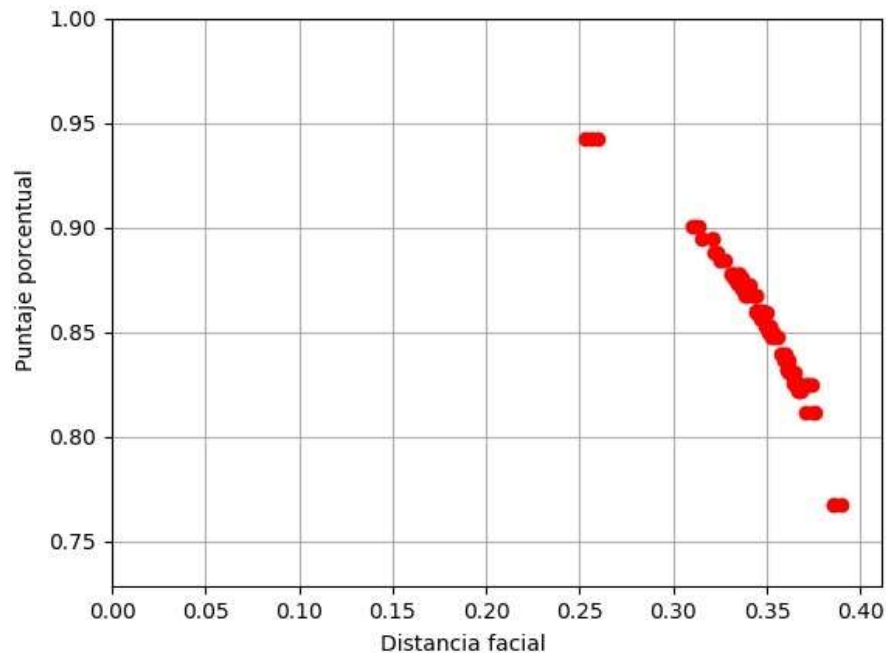


El sujeto debe mantener la cara firme durante 5 segundos; esto genera un vector con 200 predicciones de las distancias más cercanas entre todos los rostros de la base

de datos; dos rostros coinciden si su distancia es menor a 0,4 (umbral predeterminado). Para conseguir una medida de probabilidad se debe convertir el puntaje de distancia facial predicho en un puntaje porcentual de coincidencia, lo cual se demuestra en la Figura 52.

Figura 52

Puntaje de distancia entre rostros vs Puntaje porcentual



Se realiza el cálculo de la mediana de entre todas las probabilidades para obtener un único valor de probabilidad, el cual será tomado como válido, resultado del proceso se evidencia en la Figura 53.

Figura 53

Porcentaje de coincidencia a distancia 1

```

william@william-desktop: ~/Desktop/PRUEBAS/FAAM
El nombre para inscripcion es: K e v i n _ L e o n
La probabilidad de esa persona ingrese es: 0.882392883925

```

Idéntico procedimiento se realiza para el resto de sujetos a las diferentes distancias de prueba. Adicional al resultado de probabilidad se presenta el resultado del

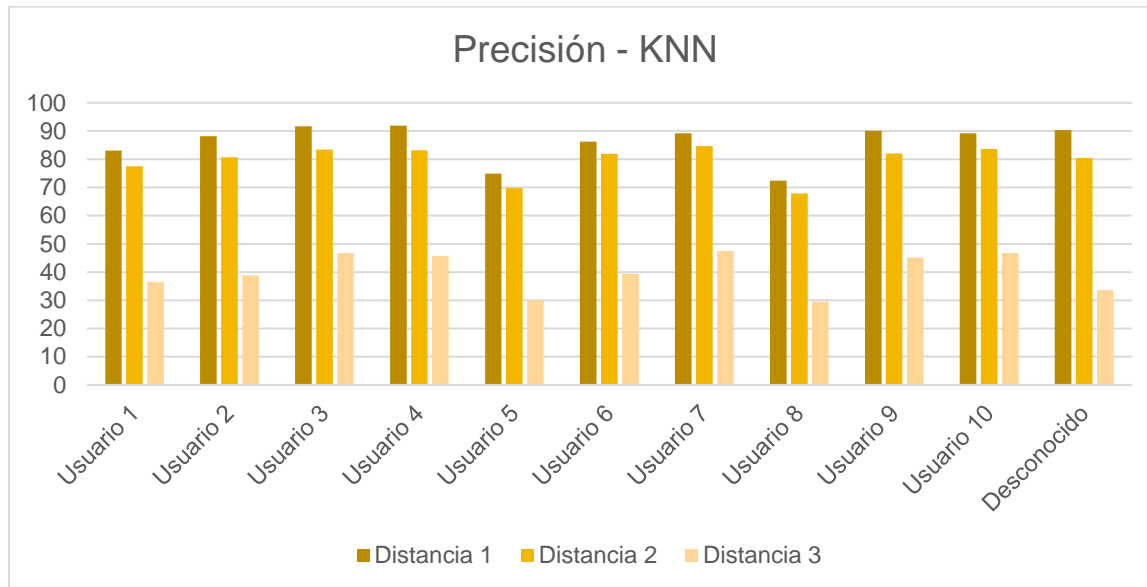
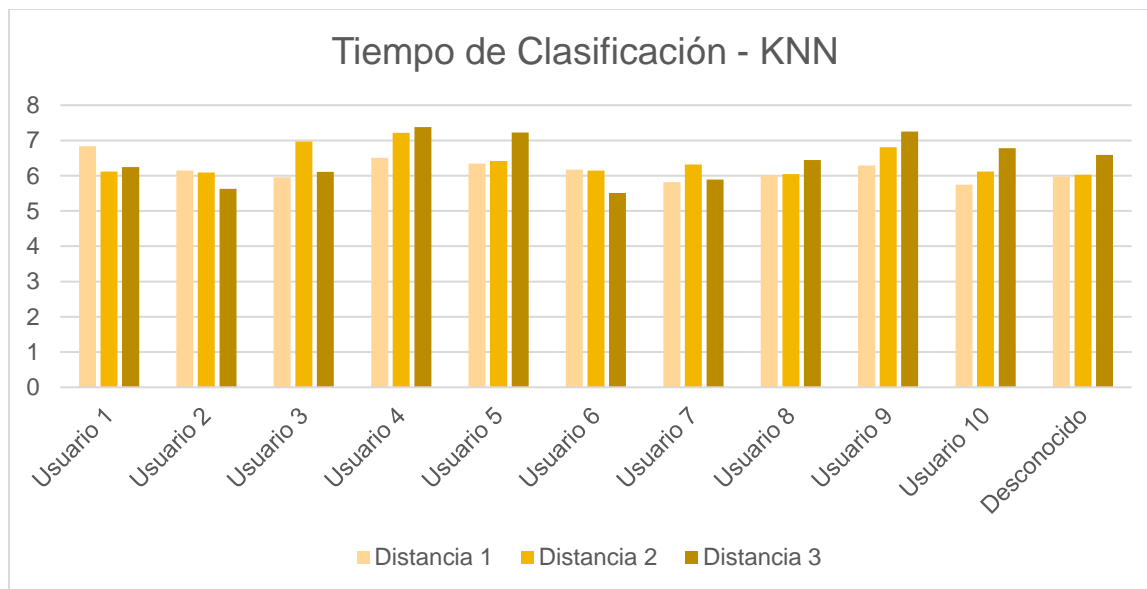
tiempo de clasificación generado por el correspondiente método de aprendizaje. Los resultados de todos los sujetos de prueba se revelan en la Tabla 19.

Tabla 19

Resultado de precisión con KNN a diferentes distancias

Sujetos	Distancia 50 cm		Distancia 80 cm		Distancia 120 cm	
	Prec [%]	Time [s]	Prec [%]	Time[s]	Prec [%]	Time [s]
Usuario 1	83,06	6,84	77,52	6,12	36,55	6,25
Usuario 2	88,14	6,15	80,68	6,09	38,81	5,63
Usuario 3	91,68	5,96	83,4	6,97	46,77	6,11
Usuario 4	91,87	6,51	83,13	7,22	45,79	7,38
Usuario 5	74,86	6,35	69,85	6,42	29,93	7,23
Usuario 6	86,25	6,17	81,90	6,15	39,39	5,51
Usuario 7	89,16	5,82	84,66	6,32	47,47	5,89
Usuario 8	72,42	6,01	67,90	6,05	29,46	6,45
Usuario 9	90,06	6,29	82,01	6,91	45,17	7,25
Usuario 10	89,21	5,75	83,61	6,12	46,82	5,78
Desconocido	90,3	5,97	80,46	6,03	33,61	6,59

En las Figuras 54 y 55 se presentan gráficos de columnas agrupadas, en los cuales se visualiza de mejor modo el contenido de la Tabla 19.

Figura 54*Precisión (%) en la prueba de Distancias***Figura 55***Tiempo de clasificación (seg) en la prueba de Distancias*

Prueba 2 – iluminación

Las Figuras 56 y 57 ejemplifican el resultado de la prueba a iluminación 1, el sujeto de prueba es el usuario 3.

Figura 56

Sujeto de prueba identificado a iluminación 1



Figura 57

Porcentaje de coincidencia a iluminación 1

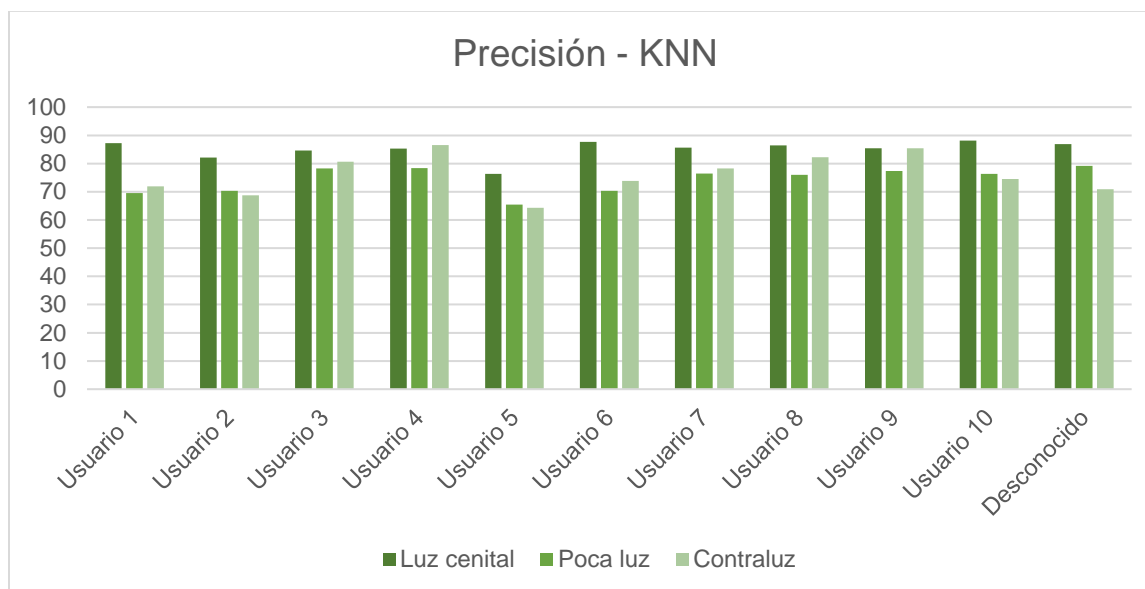
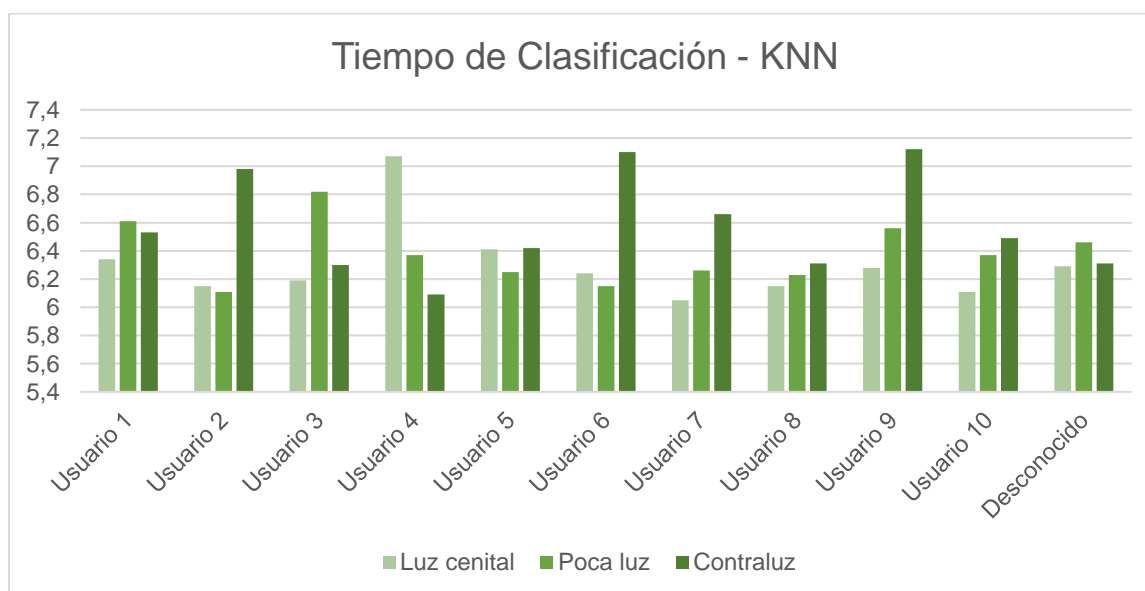
```
william@william-desktop: ~/Desktop/PRUEBAS/SVM
La probabilidad de que sea la persona es: 84.6747393476
El nombre de la persona es: William_Ibarra
El tiempo de respuesta es: 6.196629524230957
```

Para obtener el puntaje porcentual de coincidencia y el tiempo de clasificación, se lleva a cabo el mismo procedimiento explicado en la prueba de distancia. Los resultados de todos los sujetos de prueba se muestran en la Tabla 20. Cabe recalcar de la toma de datos se procedió a efectuarla manteniendo la misma medida de distancia (50 cm).

Tabla 20*Resultado de precisión con KNN a diferente intensidad de Luz*

Sujetos	Luz cenital		Poca Luz		Contraluz	
	Prec [%]	Time [s]	Prec [%]	Time [s]	Prec [%]	Time[s]
Usuario 1	87,25	6,34	69,47	6,61	71,96	6,53
Usuario 2	82,14	6,15	70,28	6,11	68,74	6,98
Usuario 3	84,67	6,19	78,26	6,82	80,59	6,3
Usuario 4	85,31	7,07	78,38	6,37	86,53	6,09
Usuario 5	76,28	6,41	65,43	6,25	64,32	6,42
Usuario 6	87,63	6,24	70,34	6,15	67,81	7,1
Usuario 7	82,61	6,05	76,44	6,16	78,71	6,66
Usuario 8	86,38	6,15	63,97	6,23	62,82	6,31
Usuario 9	84,65	6,28	77,32	6,56	85,36	7,12
Usuario 10	85,66	6,11	76,29	6,37	74,57	6,49
Desconocido	86,94	6,29	79,17	6,46	70,85	6,31

En las Figuras 58 y 59 se presentan gráficos de columnas agrupadas, en los cuales se visualiza de mejor modo el contenido de la Tabla 20.

Figura 58*Precisión (%) en la prueba de Iluminación***Figura 59***Tiempo de clasificación (seg) en la prueba de Iluminación*

Pruebas con SVM

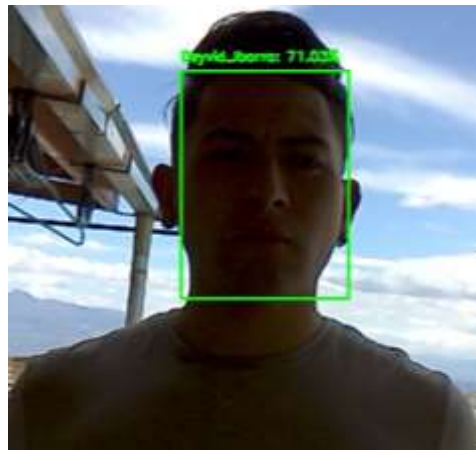
El método de aprendizaje tanto para las pruebas de distancia como para las de iluminación se configuró con los parámetros de $C=1$ y kernel = lineal.

Prueba 1 – distancia

En la Figura 60 se muestra un ejemplo de la prueba a la Distancia 3, el sujeto de prueba es el usuario 1.

Figura 60

Sujeto de prueba identificado a distancia 3




El sujeto debe mantener fijo el rostro durante 5 segundos, en este tiempo el sistema genera un vector de 200 predicciones de nombres, posteriormente el sistema decide a que rostro le corresponde mediante la mayor cantidad de veces que se repita un nombre, consiguiendo así, un valor de probabilidad de coincidencia con el sujeto de prueba.

Dos rostros coinciden si la probabilidad de coincidencia se encuentra entre 99% y 40%. La Figura 61 refleja el resultado de la predicción para el sujeto de la Figura 60.

Figura 61

Porcentaje de coincidencia a distancia 3



```

william@william-desktop: ~/Desktop/PRUEBAS/SVM
La probabilidad de que sea la persona es: 71.032864459
El nombre de la persona es: Deyvid_Ibarra
El tiempo de respuesta es: 7.349037799835205
  
```

El mismo procedimiento se realiza para el resto de sujetos a las diferentes distancias de prueba. Adicional al resultado de probabilidad se presenta el resultado del tiempo de clasificación generado por el correspondiente método de aprendizaje. Los resultados de todos los sujetos de prueba se revelan en la Tabla 21.

Tabla 21

Resultado de precisión con SVM a diferentes distancias

Distancias	Distancia 50 cm		Distancia 80 cm		Distancia 120 cm	
	Prec [%]	Time [s]	Prec [%]	Time[s]	Prec [%]	Time [s]
Usuario 1	93,09	7,93	88,72	6,59	71,03	7,34
Usuario 2	81,43	7,02	86,56	7,44	74,25	7,1
Usuario 3	93,84	6,94	85,32	7,01	72,98	6,92
Usuario 4	85,17	6,92	78,15	7,13	68,56	6,87
Usuario 5	76,95	8,41	69,42	8,17	54,14	8,84
Usuario 6	83,76	7,12	87,87	7,53	75,36	7,18
Usuario 7	91,65	7,04	83,33	7,11	71,28	7,02
Usuario 8	76,87	8,23	68,80	8,29	52,88	8,42
Usuario 9	88,81	7,02	77,33	7,23	66,96	6,93
Usuario 10	90,41	7,14	82,16	7,27	69,31	7,33
Desconocido	90,78	6,73	81,06	6,07	76,25	8,67

En las Figuras 62 y 63 se presentan gráficos de columnas agrupadas, en los cuales se visualiza de mejor modo el contenido de la Tabla 21.

Figura 62

Precisión (%) en la prueba de Distancias

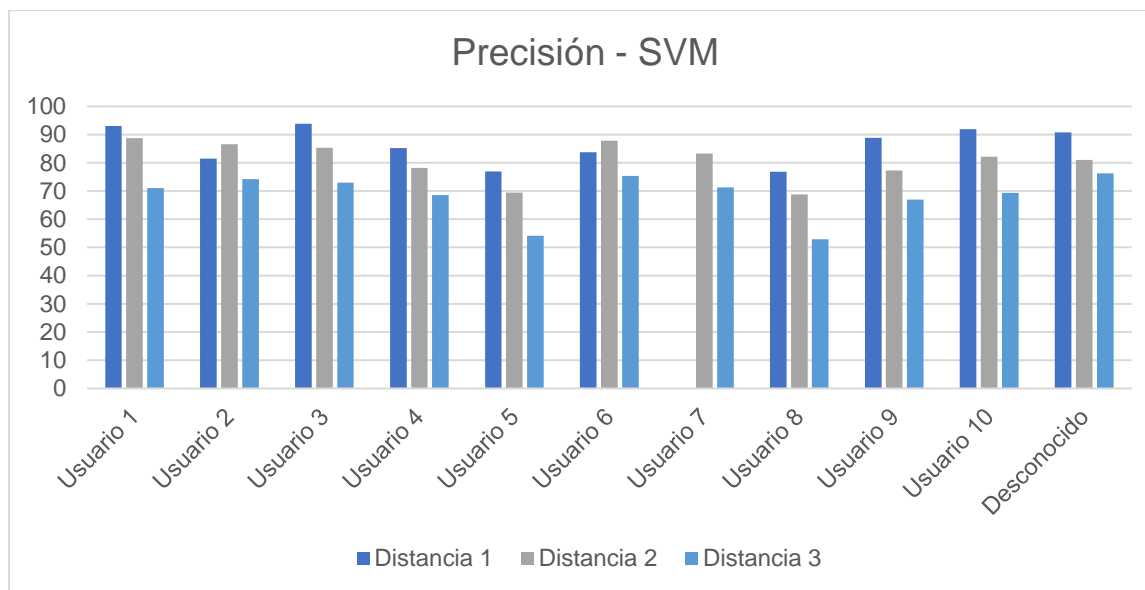
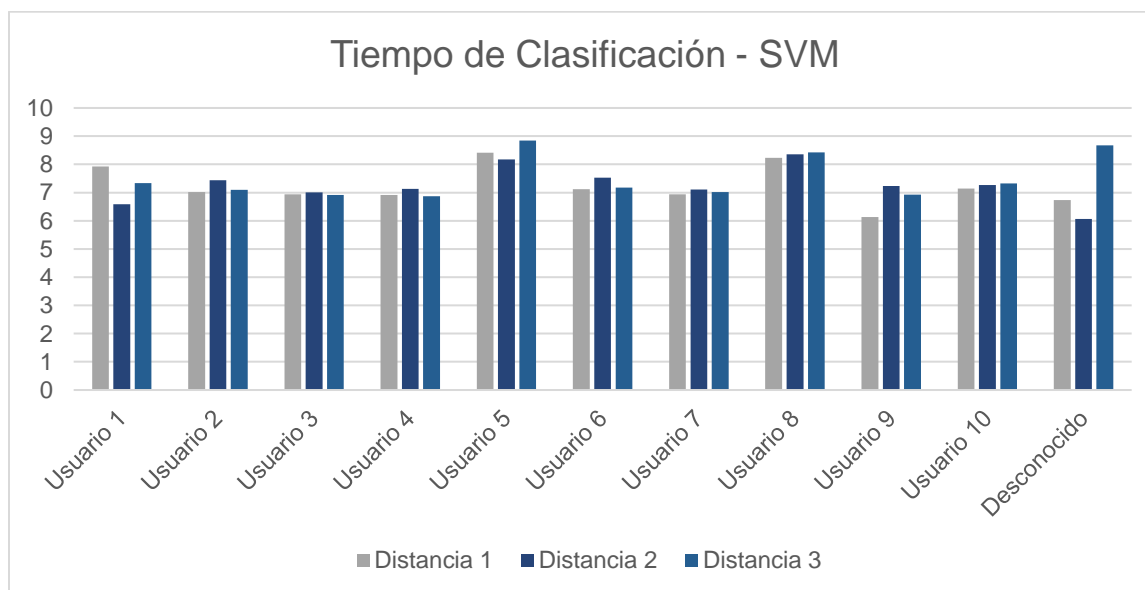


Figura 63

Tiempo de clasificación (seg) en la prueba de Distancias



Prueba 2 – iluminación

Las Figuras 64 y 65 ejemplifican el resultado de la prueba a iluminación 2, el sujeto de prueba es el usuario 2.

Figura 64

Sujeto de prueba identificado a iluminación 2



Figura 65

Porcentaje de coincidencia a iluminación 2

```
william@william-desktop: ~/Desktop/PRUEBAS/SVM
La probabilidad de que sea la persona es: 58.542749283
El nombre de la persona es: Rut_Flores
El tiempo de respuesta es: 7.635044107437134
```

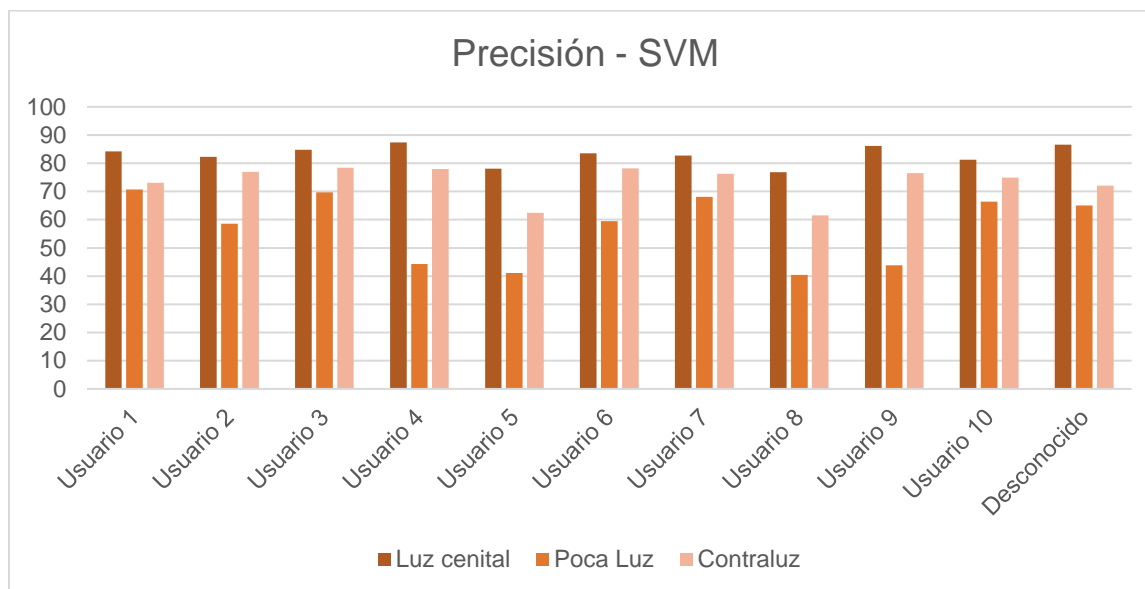
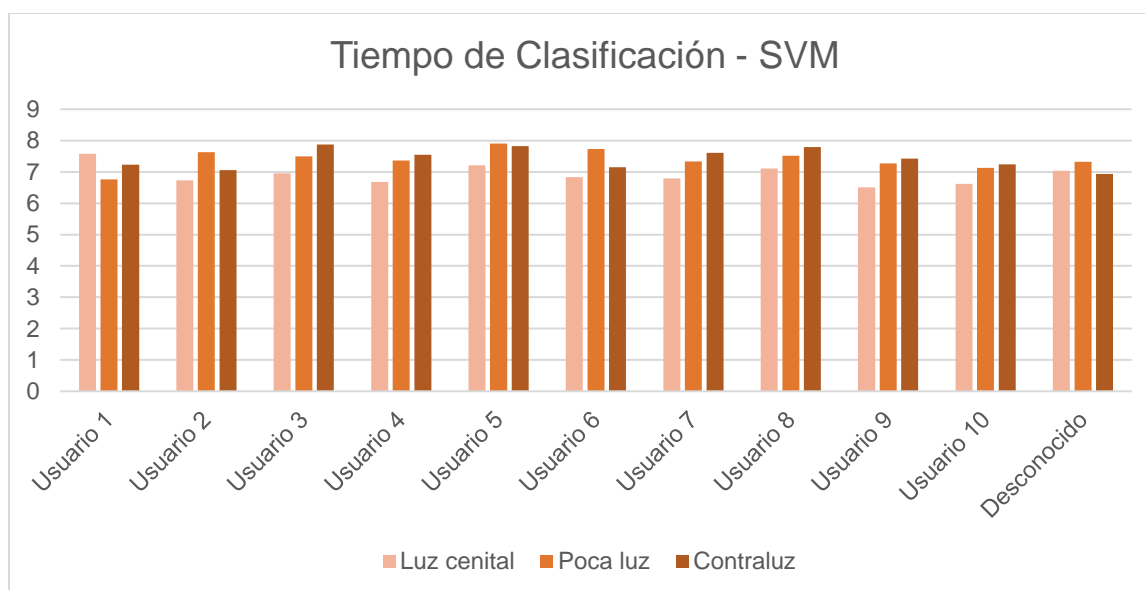
Para obtener el puntaje porcentual de coincidencia y el tiempo de clasificación, se lleva a cabo el mismo procedimiento explicado en la prueba de distancia, los resultados de todos los sujetos de prueba se muestran en la Tabla 22. Cabe recalcar de la toma de datos se procedió a efectuarla manteniendo la misma medida de distancia (50 cm).

Tabla 22

Resultado de precisión con SVM a diferentes direcciones de iluminación

Iluminación	Cenital		Poca Luz		Contraluz	
	Prec [%]	Time [s]	Prec [%]	Time[s]	Prec [%]	Time [s]
Usuario 1	84,18	7,58	70,75	6,76	73,14	7,23
Usuario 2	82,31	6,73	58,54	7,63	76,89	7,06
Usuario 3	84,81	6,96	69,67	7,5	78,41	7,87
Usuario 4	87,44	6,68	44,28	7,36	77,92	7,55
Usuario 5	78,11	7,21	41,13	7,91	62,45	7,82
Usuario 6	83,55	6,83	59,43	7,73	78,14	7,15
Usuario 7	82,74	6,79	68,12	7,33	76,28	7,61
Usuario 8	76,87	7,1	40,38	7,62	61,57	7,79
Usuario 9	86,14	6,51	43,85	7,27	76,52	7,42
Usuario 10	81,27	6,62	66,45	7,13	74,85	7,24
Desconocido	86,57	7,04	65,03	7,32	72,03	6,94

En las Figuras 66 y 67 se presentan gráficos de columnas agrupadas, en los cuales se visualiza de mejor manera el contenido de la Tabla 22.

Figura 66*Precisión (%) en la prueba de Iluminación***Figura 67***Tiempo de clasificación (seg) en la prueba de iluminación*

Análisis comparativo

Una vez registrado los resultados, se procede a realizar el respectivo análisis de rendimiento alcanzado por cada método de aprendizaje en las pruebas.

Prueba distancia

La Tabla 23 muestra el promedio de los valores obtenidos en cuanto a precisión por cada método de aprendizaje durante cada distancia de prueba.

Tabla 23

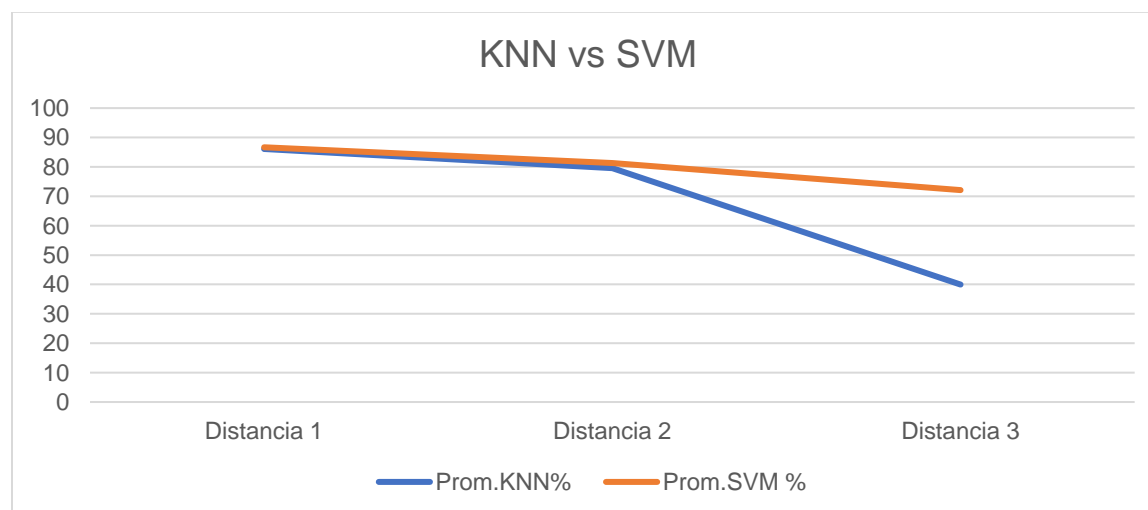
Resultados de precisión con KNN - SVM

	Distancia 1	Distancia 2	Distancia 3
Prom. KNN %	86,09	79,55	39,97
Prom. SVM %	86,67	81,33	72,1

Para ilustrar los resultados de la Tabla 23 se presenta la gráfica de la Figura 68, la cual refleja el rendimiento de cada método de aprendizaje en la tarea de clasificación a corta y mediana distancia de separación entre el sujeto de prueba y el sistema de identificación y verificación.

Figura 68

Rendimiento de cada método durante la prueba de distancias



A una distancia corta (50 cm) se distingue un comportamiento alentador en ambos métodos, la diferencia que hay entre ellos es muy estrecha, de apenas décimas. En esta distancia el método mejor posicionado es SVM, al contar con la puntuación de coincidencia de rostro más alta.

A la distancia de 80 cm, se contempla un panorama admisible para el método SVM, el cual mantiene un puntaje de coincidencia mayor al 80%. Por otro lado, en el método KNN, aunque su puntaje de coincidencia experimenta un leve descenso, podría ser considerado como aceptable.

Finalmente, a la distancia de 120 cm, se notan declives en el rendimiento de ambos métodos, siendo el método KNN el más pronunciado, llegando a obtener puntajes por debajo del 40%. En cambio, el método SVM muestra un declive moderado consiguiendo puntajes por debajo del 75%, sin embargo, hay que considerar la presencia de falsos positivos o negativos.

El tiempo de clasificación es otra métrica importante que se tomó en cuenta para el análisis, por tal motivo, en la Tabla 24 se revela los resultados de los tiempos de clasificación en promedio que alcanzó cada método de aprendizaje en la prueba de distancias.

Tabla 24

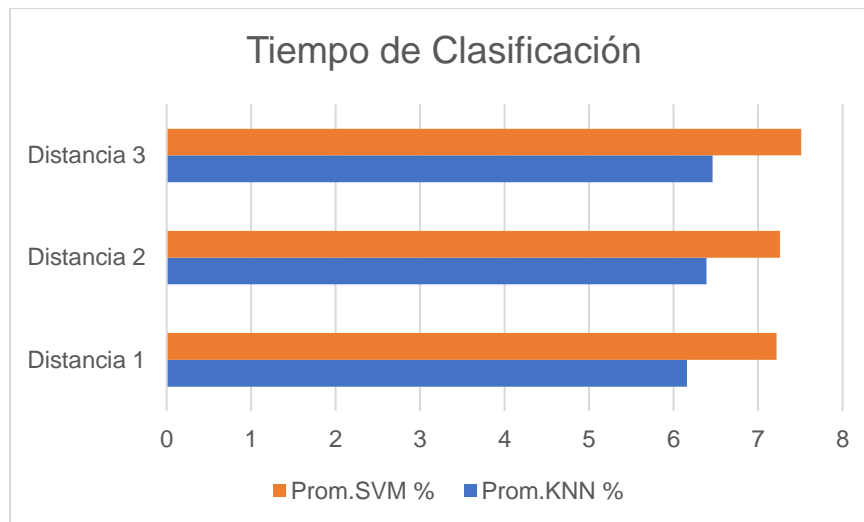
Resultados de tiempo de clasificación KNN - SVM

	Distancia 1	Distancia 2	Distancia 3
Prom. KNN [s]	6,16	6,39	6,46
Prom. SVM [s]	7,22	7,26	7,51

Una mejor perspectiva de la información contenida en la Tabla 24 se expone en el gráfico de la Figura 69.

Figura 69

Tiempo de clasificación durante la prueba de distancias



Se aprecia que para todas las distancias el método que menor tiempo de clasificación obtuvo es KNN; adicional, el incremento de tiempo de clasificación es proporcional al incremento de la distancia, es decir, mientras más lejos se encuentra el sujeto de prueba más tiempo se demorará el método en determinar la predicción.

Prueba de iluminación

La Tabla 25 presenta un promedio de los resultados de coincidencia que cada método de aprendizaje consiguió al experimentar la variación de luminosidad.

Tabla 25

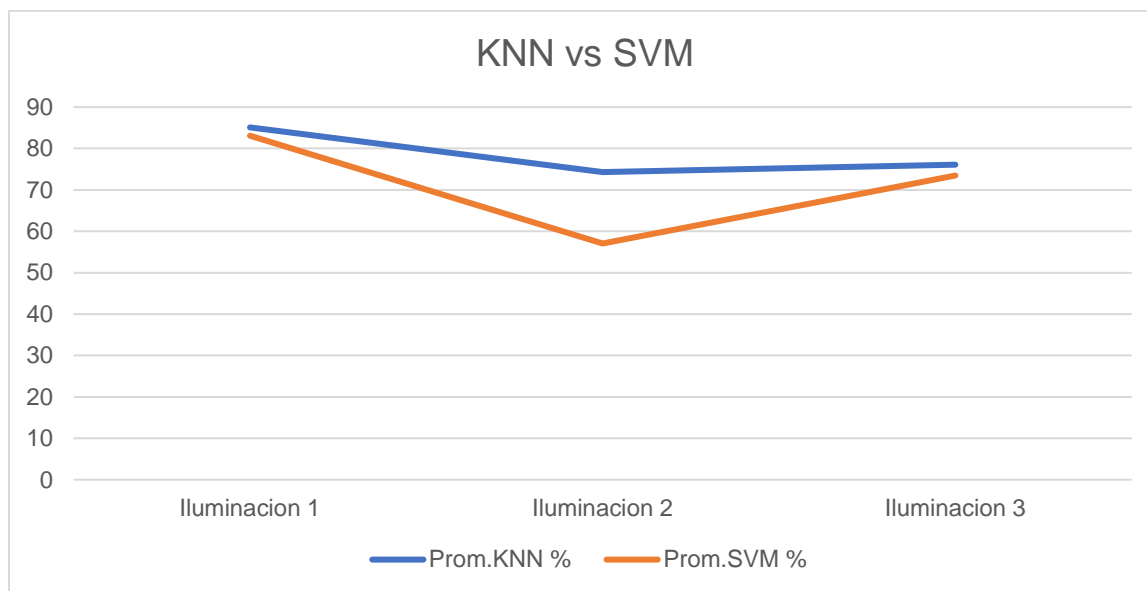
Resultados de precisión con KNN - SVM

	Iluminación 1	Iluminación 2	Iluminación 3
Prom. KNN [s]	85,10	74,3	76,11
Prom. SVM [s]	83,09	57,05	73,47

Una interpretación gráfica del contenido de la Tabla 25 se presenta en la Figura 70, exponiendo el rendimiento cada método en las diferentes intensidades de luz.

Figura 70

Rendimiento de cada método durante la prueba de iluminación



Para condiciones de iluminación en donde la luz incide sobre el rostro de manera vertical (iluminación cenital) se observa que el método que mejor puntaje de coincidencia presenta es KNN alcanzando un 85% aproximadamente.

A poca luz el método que más inconvenientes presenta en la tarea de clasificación es SVM descendiendo súbitamente su puntaje a menos del 60%, mientras que KNN percibe un descenso del 10% aprox. que podría admitirse como válido, no obstante, se contaría con la presencia de falsos positivos o negativos.

A contraluz se evidencia que los métodos tienen una mejor performance, elevando sus puntajes de coincidencia por encima del 70% aproximadamente.

Como en la anterior prueba, el tiempo de clasificación también se tomó en cuenta en la variación de iluminación revelando los resultados en la Tabla 26.

Tabla 26

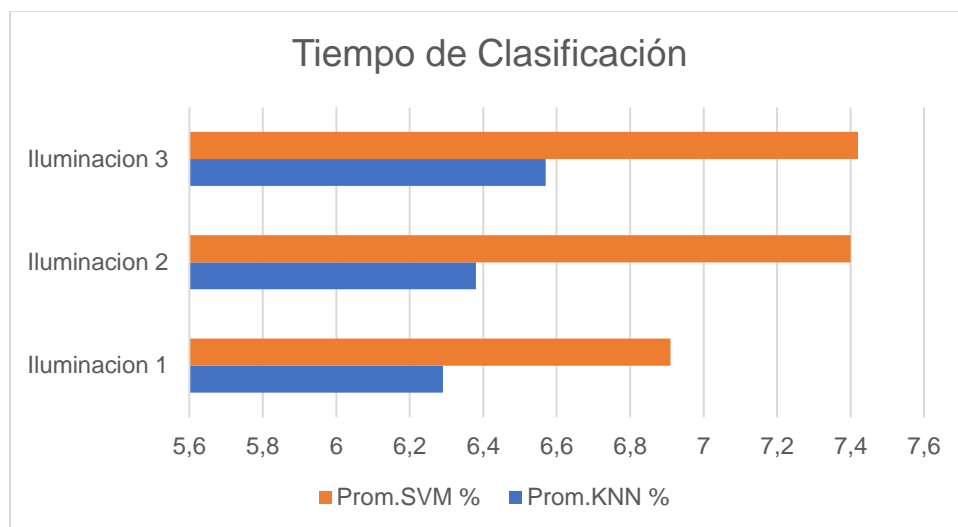
Resultados de tiempo de clasificación KNN - SVM

	Iluminación 1	Iluminación 2	Iluminación 3
Prom. KNN [s]	6,29	6,38	6,57
Prom. SVM [s]	6,91	7,40	7,42

Como descripción gráfica de los resultados de la Tabla 26 se expone la Figura 71.

Figura 71

Tiempo de clasificación durante la prueba de iluminación



En términos generales, de acuerdo a la Figura 71 se nota que el método KNN posee un mejor tiempo de clasificación que el método SVM.

Recapitulando los resultados de identificación y verificación facial, se precisa que el método SVM muestra superioridad con respecto a la variación de distancia, mientras

que el método KNN es más robusto a la variación de iluminación. En cuanto al tiempo de clasificación se evidencia que KNN es el más rápido en ambas pruebas.

Capítulo 5

Conclusiones y recomendaciones

Conclusiones

- Se desarrolló un sistema de control de acceso en tiempo real a través de reconocimiento facial y detección de tono mediante redes neuronales y dos métodos de aprendizaje automático diferentes.
- Se implementó un sistema de control de acceso en tiempo real en un dispositivo portable de baja potencia con el método de aprendizaje automático KNN, puesto que este método presenta mejores resultados para los intereses de uso del sistema.
- Los resultados de los experimentos muestran que SVM demostró ser un mejor clasificador con una precisión del 80% aprox. en promedio de acierto, en condiciones donde la distancia del sujeto de prueba es variable, mientras que KNN demostró tener mejor rendimiento en condiciones de variación de iluminación con una precisión del 78% aprox. en promedio de acierto.
- Se observó que en cuanto a velocidad de clasificación respecta, KNN demostró ser el más rápido en ambas pruebas con un tiempo de clasificación de 6 segundos aproximadamente.
- Se estructuró una base de datos con fotografías de rostros faciales de 10 personas, almacenando la información en el formato de archivo PNG ya que su cualidad de compresión sin pérdidas aporta sobremanera en el proceso de extracción de características.

- Se evidenció que tanto la cantidad como la naturaleza de la información fotográfica recolectada para la base de datos tienen un impacto significativo en los resultados, indistintamente del método de aprendizaje empleado.
- Incorporar al sistema biométrico un sistema de detección de un tono en específico, reduce significativamente los posibles intentos de suplantación de identidad.
- Se creó una base de datos que lleva el control del registro de las personas que utilizan el sistema de reconocimiento facial, el cual permite consultas en cualquier momento.
- Se presenta un sistema de control de acceso de bajo costo (en comparación con sus semejantes comerciales) integrado por los sistemas de reconocimiento facial, detección de tono y chapa magnética respectivamente.
- Es clave el uso de modelos (redes neuronales y predictores) pre-entrenados dado que permiten reducir notablemente el costo computacional y tiempo de ejecución.
- Un aspecto fundamental para limitar la detección de falsos positivos en la detección del tono es el empleo una instancia Timer que actualice constantemente la información detectada.

Recomendaciones

- En el caso de implementar el sistema de control de acceso con el método de aprendizaje automático SVM se sugiere realizar la clasificación de datos a través de un kernel rbf (radial basis function) para aumentar su rendimiento.
- Se recomienda que la base datos esté conformada por información fotográfica recolectada en diferentes condiciones de iluminación y distancia, así como también que de cada usuario se disponga de al menos 30 fotografías.

- Es importante limpiar los buffers que almacenan información de la cámara ya que eso permite que la latencia en la captura de fotogramas sea la menor posible para lograr una recepción de video en tiempo real.
- Tomar en cuenta la potencia de alimentación que debe ser suministrada a la tarjeta de desarrollo, puesto que con una alimentación menor a 10W el sistema no podrá entrar en funcionamiento.
- Es preciso el equipamiento de un sistema de refrigeración para prevenir futuros daños y prolongar la vida útil de la tarjeta de desarrollo.
- Conviene configurar el micrófono USB con una sensibilidad del 85% para disminuir la probabilidad de detección de falsos positivos.
- Se sugiere el empleo de una cámara CSI por disponer de una mayor velocidad que transmisión de datos en comparación con una de USB 3.0 o una IP, así como también que la cámara disponga de visión multimodo (diurna/nocturna).

Trabajos futuros

Como trabajo futuro se plantea la inclusión de un segundo método de control de acceso mediante la implementación de algoritmos de procesamiento digital de señales de voz con aprendizaje automático y redes neuronales para el reconocimiento de voz, asegurando así mayor seguridad.

Además, también se propone presentar la información de la base de datos que lleva el control del registro de ingreso mediante un sitio de consulta web o una App.

Referencias bibliográficas

- Agatonovic, S., & Beresford, R. (2000). Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. *Journal of Pharmaceutical and Biomedical Analysis*, 717–727.
- Arel, I., Rose, C., & Karnowski, T. (2010). Deep machine learning — a new frontier in artificial intelligence. *IEEE Computational Intelligence Magazine*, 5:13–18.
- Baird, S. L. (2002). "Biometrics 'security technology': it is important for students to understand that technology can be used as part of a solution to a problem. *Resources In Technology*, 18.
- Belhumeur, P. (1997). *Eingenfaces vs fisherfaces: Recognition using class specific linear projection* . IEEE Transactions on Pattern Analysis and Machine Intelligence .
- Bledsoe, W. (1968). *Semiautomatic facial recognition*. California: Technical report sri project 6693, Stanford Research Institute.
- Bolle, Connel, Pankanti, & Senior, R. &. (2004). *Guide to Biometrics*. New York: Springer-Verlag.
- Bradski, G. (1998). *Computer vision face tracking for use in a perceptual user interface*. Intel Technology Journal.
- Castro, V. (2018). *Diseño de un sistema de identificación de personas* . Burgos : Escuela Politécnica Superior.
- Clarke, R. (1994). Human identification in information systems: Management challenges and public policy issues. *Information Technology & People*, 7(4):6-37.

- Constante Prócel, P. G. (2016). *Artificial Vision Techniques for Strawberry's Industrial Classification*. Sangolquí: IEEE Latin America Transactions, 14, 2576 - 2581.
- Cortés, J., Ardila, W., & Mendoza, J. (2011). *Técnicas alternativas para la conversión de imágenes a color a escala de grises en el tratamiento digital de imágenes*. No. 47, pp. 207–212.
- Cover, T., & Hart, P. (1967). *Nearest neighbor pattern classification*. IEEE Transactions on Information Theory, vol. 13, no. 1, pp. 21-27.
- Cubero, S. (2012). *Diseño e implementación de nuevas tecnologías basadas en visión artificial para la inspección no destructiva de fruta en campo y mínimamente procesada*. Valencia: Instituto valenciano de investigaciones agrarias.
- Datal, J., & Patel, S. (2013). *Instant OpenCV Starter*. . Packt Publishing. .
- Davies, R., Twining, C., & Taylor, C. (2008). *Statistical Models of Shape*. London: Springer-Verlag.
- Dharaiya, D. (2020). *History of Facial Recognition Technology and its Bright Future*. <https://readwrite.com/2020/03/12/history-of-facial-recognition-technology-and-its-bright-future/>.
- Draghici, S. (1997). *A Neural Network Based Artificial Vision System for Licence Plate Recognition*. Detroit : International Journal of Neural Systems.
- Fernández, A. (2019). *Mastering OpenCV 4 with Python: A practical guide covering topics from image processing, augmented reality to deep learning with OpenCV 4 and Python 3.7*. Birmingham: Packt Publishing.
- Geitgey, A. (2018). *Face recognition* . https://github.com/ageitgey/face_recognition.

- Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). *Rich feature hierarchies for accurate object detection and semantic segmentation*. Columbus: Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition.
- González, R., Barrientos, A., Toapanta, M., & Del Cerro, J. (2017). *Aplicación de las Máquinas de Soporte Vectorial (SVM) al diagnóstico clínico de la Enfermedad de Pákinson y el Temblor Esencial*. Revista Iberoamericana de Automática e Informática Industrial .
- Hornberg, A. (2017). *Handbook of Machine and Computer Vision*. Wiley-VCH Verlag GmbH & Co. KGaA.
- Introna, L., & Nissenbaum, H. (2009). *Facial Recognition Technology: A Survey of Policy and Implementation Issues*.
- Jain, A., Ross, A., & Prabhakar, S. (2004). An Introduction to Biometric Recognition. *IEEE transactions on circuits and systems for video technology*.
- King, D. (2016). *Max-Margin Object Detection*.
- Kirby, M., & Sirovich, L. (1988). *Application of the Karhunen-Loeve Procedure for the Characterization of Human Faces*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, No. 1, pp. 103-108.
- Kotsiantis, S. (2007). *Supervised Machine Learning: A Review of Classification Techniques*. Greece: University of Peloponnese.
- Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). *Imagenet classification with deep convolutional neural networks*. Lake Tahoe: Proceedings of the 26th Annual Conference on Neural Information Processing Systems.

- Kumar, P. G. (2017). *Intelligent face recognition and navigation system using neural*. India: Springer Science+Business Media, LLC.
- Lucas, M. (1997). *Face recognition with the continuous n-tuple classifier*. British Machine vision conference.
- Marqués, I. (2010). *Face Recognition Algorithms*. Universidad del País Vasco.
- Matich, D. (2001). *Redes Neuronales: Conceptos Básicos y Aplicaciones*. Grupo de Investigación Aplicada a la Ingeniería Química (GIAIQ).
- Miller, B. (1994). Vital signs of identity. *IEEE Spectrum*, 31(2):22-30.
- Mordvintsev, A. (2017). *OpenCV-Python Tutorials Documentation Release 1*.
- Obando, I. (2019). *Implementación de un control de acceso biométrico mediante reconocimiento facial*. Sangolquí.
- Paolini, G., Hernández, A., & Pereyra, V. (2018). *Frecuencia fundamental de habla de voz normal según sexo*. Córdoba, Argentina.
- Raducanu, B., & Vitria, J. (2008). *Face Reconginiton by Artificial Vision Systems*. World Scientific Publishing Company: International Journal of Pattern Recognition.
- Rajasegaran, J., Jayasundara, V., & Jayasekara, S. (2019). *DeepCaps: Going Deeper with Capsule Networks*. Computer Vision Fundation .
- Rajeshwar, D., Ritu, R., & Dharmender, K. (2012). *Face Recognition Techniques: A Review*. International Journal of Engineering Research and Development, Volume 4, Issue 7, PP. 70-78.
- Ranz, J. R. (2008). *Algoritmos genéticos aplicados a la segmentación de imágenes con iluminación no controlada*. Madrid.

Anexos