Reconocimiento de expresiones faciales y características personales como herramienta para identificar personas en un sistema de transporte público

Uwe Rojas Villanueva
https://orcid.org/0000-0003-0423-1001

Juan Carlos Goñi Delión
https://orcid.org/0000-0001-8855-9543

Fabricio Paredes Larroca
https://orcid.org/0000-0001-8857-9253

Universidad de Lima, Facultad de Ingeniería y Arquitectura, Lima, Perú

Recibido: 21 de mayo del 2021 / Aprobado: 30 de junio del 2021
doi: https://doi.org/10.26439/ing.ind2022.n.5811

RESUMEN. La inteligencia artificial en la actualidad tiene muchas aplicaciones. En este artículo se plantea el reconocimiento facial basado en inteligencia artificial usando *machine learning* para identificar, a través de lenguaje Python, a personas que se encuentran extraviadas, raptadas o que han cometido delitos. La plataforma de desarrollo Jetson Nano identifica y envía una alerta a través de un mensaje de texto SMS a las unidades de supervisión y control de información para la toma de decisión y respuesta. Este dispositivo funciona con el sistema operativo Ubuntu, que tiene la capacidad de trabajar en forma autónoma (*standalone*), es pequeño y de fácil accesibilidad en espacios reducidos. Asimismo, la herramienta puede predecir el estado de ánimo de las personas a través de gestos realizados en el rostro con la aplicación del algoritmo de Viola-Jones.

PALABRAS CLAVE: inteligencia artificial / reconocimiento facial (informática) / algoritmos computacionales / aprendizaje automático

Correos electrónicos en orden de aparición: urojas@ulima.edu.pe, jgoni@ulima.edu.pe, fparedes@ulima.edu.pe

RECOGNITION OF FACIAL EXPRESSIONS AND PERSONAL FEATURES AS A TOOL TO IDENTIFY PEOPLE IN A PUBLIC TRANSPORT SYSTEM

ABSTRACT. The use of artificial intelligence nowadays has many applications. This paper proposes the use of artificial intelligence based facial recognition using machine learning to identify through Python language, people who are missing, abducted or have committed crimes. The Jetson Nano development platform identifies and sends an alert via SMS text message to the monitoring and information control units for decision making and response. This device is based on the Ubuntu operating system, which can work standalone, is small, and allows easy accessibility in confined spaces. The tool can also predict people's moods through gestures made on the face with the application of the Viola-Jones algorithm.

KEYWORDS: artificial intelligence / human face recognition (computer science) / computer algorithms / machine learning

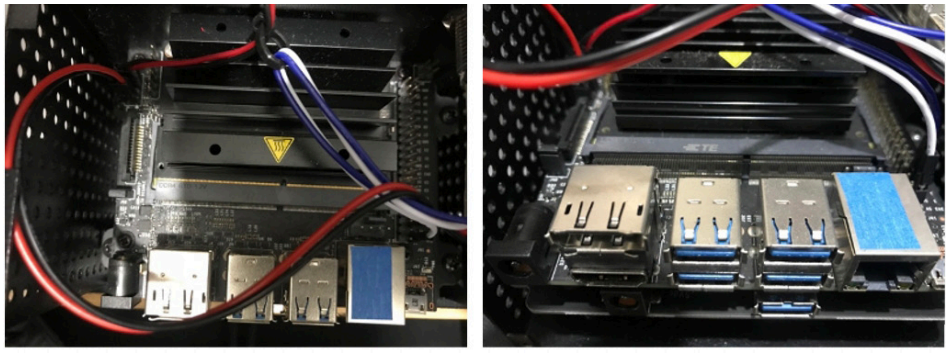
INTRODUCCIÓN

En la actualidad, está tomando importancia la identificación de rostros de personas con cámaras instaladas en las calles de las ciudades, debido a circunstancias adversas para la población como la inseguridad ciudadana y la necesidad de identificar personas, analizar sus estados de ánimo y otros. Este trabajo propone el uso de una tecnología que permita hacer estudios de forma masiva en aquellos lugares donde las cámaras o servicios del Estado no son capaces de tener algún tipo de acceso. Para esta aplicación se ha realizado un reconocimiento biométrico (Duró, 2001) a través de un trabajo multidisciplinario, utilizando metodologías que permitan obtener reconocimiento de patrones para clasificar rostros (Gualdrón, 2013). Este tipo de aplicaciones identifica rostros de enojo, cansancio, miedo, felicidad, tristeza, sorpresa y normalidad (Andrago Calvachi, 2019).

La inteligencia artificial se ha convertido en una herramienta indispensable en nuestra sociedad y puede ayudar a mejorar el bienestar humano (Estévez Martín & Ramírez Barredo, 2018). Para la realización del trabajo se utiliza el dispositivo Jetson Nano, de la compañía NVIDIA, catalogada como líder en computación de inteligencia artificial. La figura 1 muestra el procesador Jetson Nano, que contiene cinco puertos USB, un conector RJ45 y un conector HMI. Jetson Nano es un dispositivo electrónico que puede ejecutar una variedad de librerías de aplicación para *machine learning* utilizando tecnologías como TensorFlow, PyTorch, Caffe/Caffe2, Keras, MXNet y otros.

Figura 1

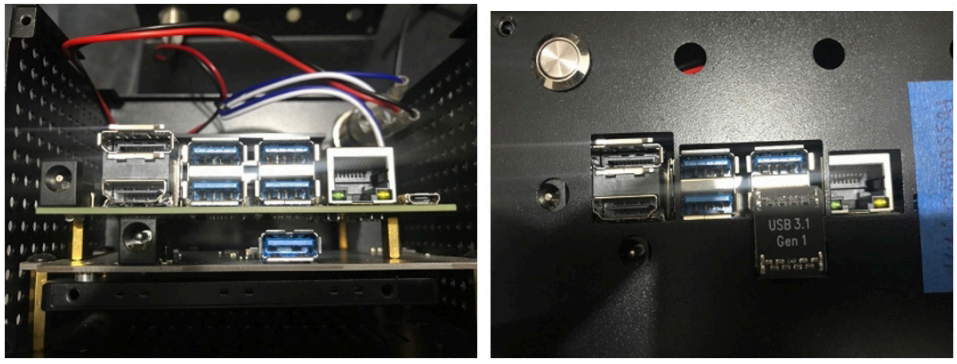
Procesador Jetson Nano, con cinco puertos USB, un conector RJ45 y un conector HMI



Una de las principales aplicaciones de Jetson Nano se encuentra en el reconocimiento de imágenes, la detección y localización de objetos; actualmente, se utiliza en seguridad ciudadana para la identificación de personas, en el entretenimiento, entre otros fines. Jetson Nano es capaz de trabajar por más de diez horas, alimentado por una batería de 18 V y 3000 mA de polímero de litio. En la figura 2, se observa el dispositivo Jetson Nano con extensión de disco SSD 240 GB, conectada por USB 3.1.

Figura 2

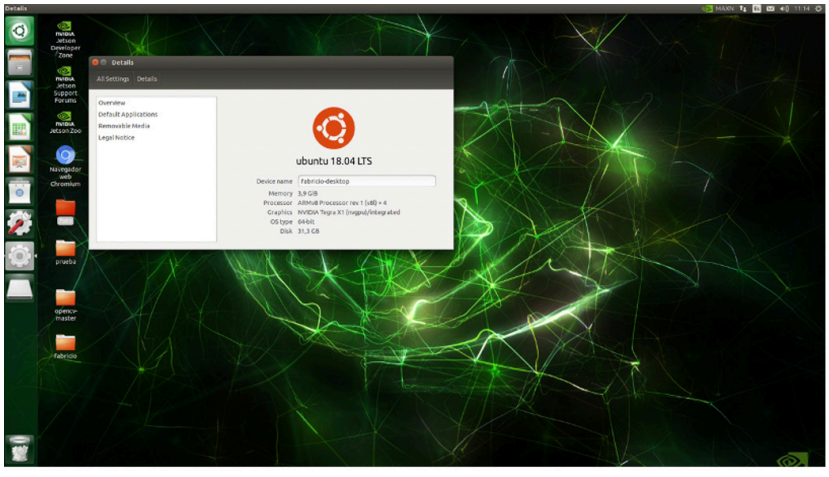
Jetson Nano con extensión de disco SSD 240gb, conectada por USB 3.1



Para la programación de Jetson Nano, el sistema operativo está basado en Ubuntu 18.04 LTS (véase la figura 3). El algoritmo propuesto es el algoritmo de detección Viola-Jones, que es un sistema capaz de detectar el estado de ánimo en la cara de una persona, para lo cual usa una cámara de video. En el proceso, la cámara captura una imagen y el *software* implementado con el algoritmo de Viola-Jones genera un conjunto de fotogramas con un identificador particular igual a 21 fotos en escala de grises; con ello crea una base de datos que luego servirá para poder hacer la inferencia del análisis deseado.

Figura 3

Sistema operativo en Jetson Nano, Ubuntu 18.04 LTS



En una segunda etapa, se utiliza un *software* implementado con lenguaje de programación Python, que usa la cámara web para comparar una imagen actual con

las imágenes contenidas en la base de datos. Luego, por métodos de entrenamiento y aprendizaje (*machine learning*), se obtiene un *software* capaz de predecir un conjunto de imágenes en tiempo real.

Este mecanismo de identificación facial se deberá implementar en un sistema de transporte público masivo en la ciudad de Lima, principalmente en el servicio del Metropolitano, donde cada una de las puertas de acceso de las estaciones de ingreso y salida contará con un dispositivo de reconocimiento facial como el planteado en este trabajo.

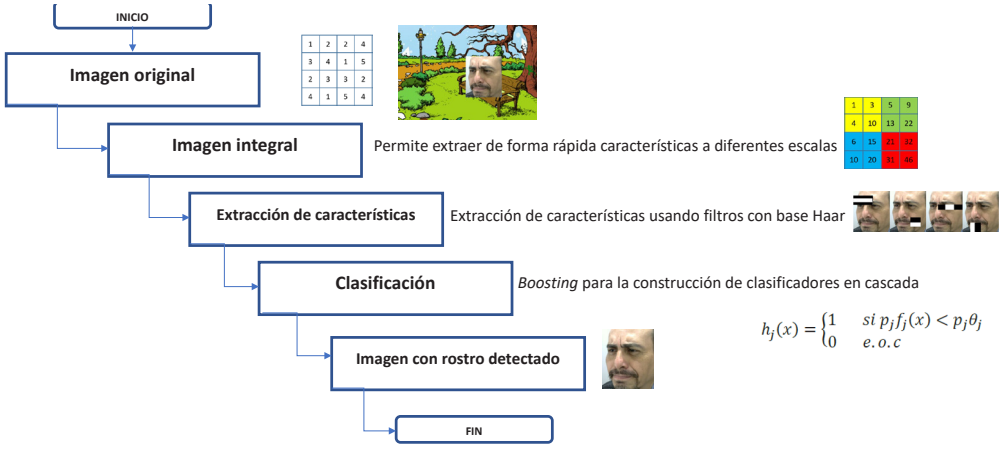
MARCO TEÓRICO

Algoritmo de detección de Viola-Jones

El algoritmo de Viola-Jones (Viola et al., 2005) es un algoritmo de detección de las características más relevantes de un rostro humano mediante el uso de filtros (véase la figura 4), que se asemejan a las funciones base de las transformadas características Haar-like (véase la figura 5; Lienhart & Maydt, 2002) que se presentan como nuestro kernel convolucional utilizado en SVM (*support vector machine*). Estos filtros se aplican sobre la imagen y dan la posibilidad de obtener muchos resultados llamados *clasificadores*. Sin embargo, estos usualmente no son útiles por su baja eficiencia; entonces, se aplica el algoritmo AdaBoost (Wei et al., 2004), cuya función es tomar clasificadores débiles que no son útiles para estos fines y combinarlos para construir un clasificador fuerte con características relevantes. El algoritmo de Viola-Jones emplea una "cascada de clasificadores", que básicamente es un árbol de decisión en cascada.

Figura 4

Algoritmo de Paul Viola y Michael Jones



Haar-like

En el trabajo presentado por Viola-Jones existen tres tipos para la extracción de características (*features*). La figura 4 muestra ejemplos de *features* de dos, tres y cuatro rectángulos, y su posición relativa a la ventana de búsqueda (Lienhart & Maydt, 2002). La suma de los píxeles en las áreas grises se resta de la de las áreas blancas. Los *features* son los siguientes:

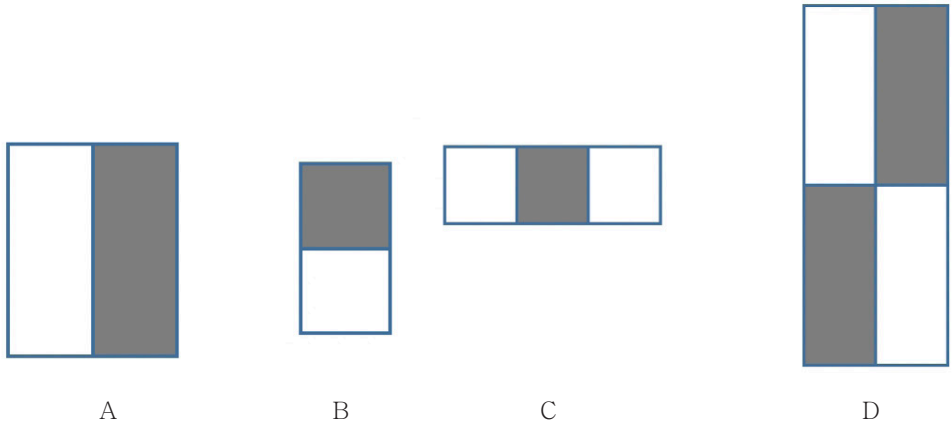
a. *Features* de dos rectángulos cuyo valor es la diferencia entre las sumas de los píxeles contenidos en ambos rectángulos. Las regiones tienen la misma área y forma, y son adyacentes (véase la figura 5, A, B).

b. *Features* de tres rectángulos que calculan la diferencia entre los rectángulos exteriores y el interior multiplicado por un peso para compensar la diferencia de áreas (véase la figura 5, C).

c. *Features* de cuatro rectángulos que computan la diferencia entre pares diagonales de rectángulos (véase la figura 5, D).

Figura 5

Features A y B de dos rectángulos, C de tres rectángulos, D de cuatro rectángulos y posición relativa a la ventana de búsqueda



Nota. De "An Extended Set of Haar-Like Features for Rapid Object Detection", por R. Lienhart y J. Maydt, en *Proceedings. International Conference on Image Processing* (vol. I), 2002 (DOI: 10.1109/ICIP.2002.1038171). Derechos de autor 2002 IEEE.

Imagen integral

Una imagen integral es la suma de los píxeles de un rectángulo, la cual puede ser calculada de manera muy eficiente empleando una representación intermedia denominada

imagen integral. La imagen integral en el punto (*x*, *y*) contiene la suma de todos los píxeles que están arriba y hacia la izquierda de ese punto en la imagen original (véase la figura 6).

Figura 6

Imagen integral a partir de una imagen original

1	2	2	4	1
3	4	1	5	2
2	3	3	2	4
4	1	5	4	6
6	3	2	1	3

Imagen original

0	0	0	0	0	0
0	1	3	5	9	10
0	4	10	13	22	25
0	6	15	21	32	39
0	10	20	31	46	59
0	16	29	42	58	74

Imagen integral

$$ii(x,y) = \sum_{x' \le x,\, y' \le y} i(x',y') \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

La descripción matemática (2.1) de la imagen integral proveniente de una imagen (imagen original). La imagen integral se puede calcular en un solo barrido de la imagen empleando el siguiente par de sentencias recurrentes:

Para una matriz de orden 3:

1	2	2
3	4	1
2	3	3

(i)

1	3	5
4	10	13
6	15	21

(ii)

Para matriz (i):

$$s(x=0,y=0) = s(x=0,y=-1) + i(x=0,y=0) = 1 \text{ (suma de fila)}$$
$$s(x=0,y=1) = s(x=0,y=0) + i(x=0,y=1) = 3 \text{ (suma de fila)}$$
$$s(x=1,y=0) = s(x=1,y=-1) + i(x=1,y=0) = 3 \text{ (suma de fila)}$$
$$s(x=1,y=1) = s(x=1,y=0) + i(x=1,y=1) = 7 \text{ (suma de fila)}$$

Generalizando:

$$s(x,y) = s(x,y-1) + i(x,y) \quad \text{.....(2.2)}$$

Para matriz (i):

$$ii(x=0,y=1) = i(x=0,y=0) + i(x=0,y=1) = 3$$

$$ii(x=1,y=0) = i(x=0,y=0) + i(x=1,y=0) = 4$$

$$ii(x=1,y=1) = i(x=0,y=0) + i(x=0,y=1) + i(x=1,y=0) + i(x=1,y=1) = 10$$

Generalizando:

$$ii(x,y) = ii(x-1,y) + s(x,y) \quad \text{.....(2.3)}$$

Se observa que (x, y) es la suma acumulada de la fila x, con $s(x,-1) = 0$ y $ii(-1,y) = 0$.

Clasificación: proceso de aprendizaje

Es necesario realizar un proceso de entrenamiento supervisado para crear la cascada de clasificadores. Este proceso se lleva a cabo mediante un algoritmo basado en AdaBoost (Zerrouki et al., 2018), un metaalgoritmo adaptativo de *machine learning* cuyo nombre es una abreviatura de *adaptive boosting*. El *boosting* consiste en tomar una serie de clasificadores débiles y combinarlos para construir un clasificador fuerte con la precisión deseada. AdaBoost fue introducido por Freund y Schapire en 1995 para resolver muchas de las dificultades prácticas asociadas al proceso de *boosting*.

En el procedimiento de Viola-Jones, AdaBoost se utiliza tanto para seleccionar un pequeño set de *features* de las 180 000 posibles como para entrenar el clasificador. Para seleccionar *features*, se entrenan clasificadores débiles limitados a usar una única *feature*. Para cada *feature*, el clasificador débil determina el valor umbral que minimiza los ejemplos mal clasificados. Un clasificador débil $h_j(x)$, por tanto, consiste en una *feature* f_j, un valor umbral Θ_j y un coeficiente p_j; la dirección la indica el signo de desigualdad.

$$h_j(x) = \begin{cases} 1 & si\ p_j f_j(x) < p_j \theta_j \\ 0 & e.o.c \end{cases}$$

Algoritmo AdaBoost

1. Se parte de un conjunto de imágenes $(x_1, y_1), \ldots, (x_n, y_n)$, donde $y_i = 0, 1$ para ejemplos negativos y positivos respectivamente.
2. Se inicializan los pesos $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ para $y_i = 0, 1$ respectivamente, donde m es el número de negativos y l es el número de positivos.

3. Para cada ronda, $t = 1, \ldots, T$:

   3.1 Normalizar los pesos: $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$

   3.2 Para cada *feature j*, entrenar un clasificador h_j que solo use una *feature*. El error se evalúa teniendo en cuenta los pesos w_t, $\epsilon_j = \sum_i \quad w_i |h_j(x_i) - y_i|$

   3.3 Se escoge el clasificador h_t, con menor error ϵ_t.

   3.4 Se actualizan los pesos: $w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$, donde $e_i = 0$ si el ejemplo x_i se clasifica correctamente y 1 en caso contrario; y $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$

El clasificador fuerte final es este:

$$h(x) = \left\{1 \; si \; \sum_{t=1}^{T} \quad \alpha_t h_t(x) \geq \frac{1}{2}\sum_{t=1}^{T} \quad \alpha_t \; 0 \; e.o.c.\right\}, \text{ donde: } \alpha_t = \frac{1}{\beta_t}$$

METODOLOGÍA

Para este estudio se implementó una aplicación que utiliza una plataforma de desarrollo Jetson Nano, del fabricante NVIDIA. Jetson Nano es una robusta minicomputadora portátil con una gran potencia de procesamiento y, a la vez, de bajo consumo de energía: trabajando a su máxima potencia, es 10 vatios. Tiene la capacidad de trabajar con procesamiento paralelo para realizar la clasificación de objetos obtenidos con una cámara web; también puede procesar audio. El sistema operativo se encuentra embebido en una tarjeta mini-SD de alta velocidad, bajo la plataforma de Linux con una integración de 128 *core* Maxwell GPU en un procesador *quad core* ARM A57 de 64 bits, con 4 GB de memoria tipo LPDDR4. Esto permite optimizar sustancialmente el tiempo de desarrollo de la aplicación y los costos de horas inmersos en el proyecto. Una de las principales características es el uso de *frameworks*, como TensorFlow, Visionworks y CUDA Toolkit. Adicionalmente, posee entradas y salidas digitales para poder recibir señales y generar salidas al mundo exterior. Asimismo, tiene la posibilidad de generar dos señales PWM para uso de servomotores o motores de DC. La cámara es una *webcam* modelo C505 HD de *logitech* de 720 píxeles con velocidad de captura de 30 *frames* por segundo o FPS.

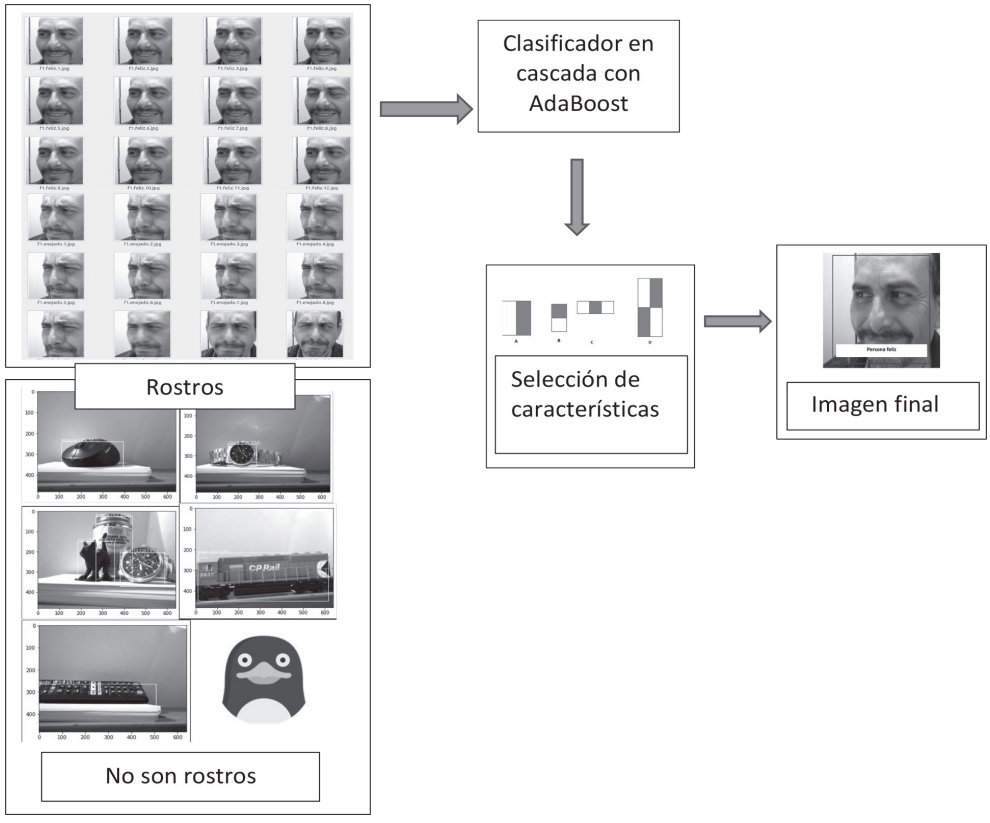
En la aplicación planteada, se usa la programación en Python a través de las librerías OpenCV2, CV2 para la detección, análisis y tratamientos de imágenes mediante algoritmos de inteligencia artificial. Otra de las herramientas utilizadas es el *numpy*, que es una potente estructura de datos que implementa matrices multidimensionales que garantizan los cálculos eficientes entre las matrices. Estos paquetes son indispensables para el uso de *machine learning.*

El algoritmo de Viola-Jones —visto en el marco teórico— tiene una probabilidad de éxito del 99,9 % y una probabilidad de falla del orden del 3,33 %, donde solo procesa

la información de la imagen en escala de grises. Lo que usa de la imagen es la llamada *imagen integral* para determinar si en una imagen se encuentra una cara. En la figura 7, se observa el esquema del motor de inferencia del algoritmo de Viola-Jones. Este algoritmo divide la imagen integral de prueba en subregiones de tamaños diferentes y las utiliza como una serie de clasificadores "cascada". El ahorro computacional de tiempo en detectar la cara según sus características personales es considerablemente bajo, ya que no se procesan las regiones de la imagen que no contienen un rostro facial presente.

Figura 7

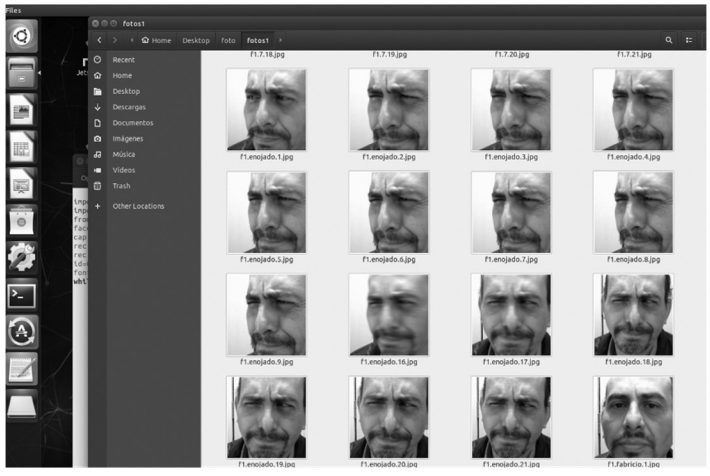
Implementación del algoritmo de Viola-Jones en la aplicación



Un ejemplo de la base de datos obtenida se presenta en la figura 8. El *dataset*, en este caso, se consigue a través de la cámara web frente a la persona, que captura diferentes instantes. En total se tomaron 21 capturas o fotogramas, creando un archivo llamado trainingdata.yml.

Figura 8

Fotogramas de la captura de gestos con expresión enojada



Teniendo una base de datos muy amplia de imágenes de rostros y de estados de ánimo, se realiza el entrenamiento del algoritmo con las imágenes, seleccionando las características necesarias de los gestos faciales para el funcionamiento adecuado de los clasificadores en cascada. La figura 9 muestra el *dataset* de gestos con expresión de felicidad.

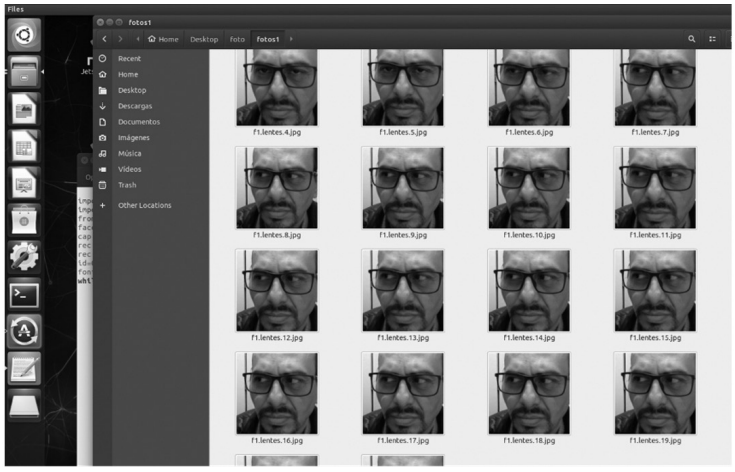
Figura 9

Fotogramas de la captura de gestos con expresión de felicidad

De la misma manera, en la figura 10 se presenta una base de datos en la que se destaca una particularidad especial: incluye los lentes de una persona en su rostro. La persona puede estar enojada, o triste o inmutable, pero el algoritmo en cascada ha sido preparado para reconocer esta peculiaridad.

Figura 10

Fotogramas de la captura de cara con lentes



Las figuras expuestas en los resultados corresponden a la base de datos que permite hacer la inferencia en el sistema gestual.

RESULTADOS

Los resultados que se describen a continuación muestran la obtención de algunas características propias de las personas identificadas con algunos rasgos significativos. Con el desarrollo y la mejora del *software* se podría conseguir una mejor fusión de las características.

El sistema propuesto es capaz de identificar una foto exhibida desde un celular frente a la cámara web e indicar su estado de ánimo o algunos rasgos significativos, como se observa en el siguiente ejemplo. La figura 11 es la foto en cámara web de un rostro frente a un teléfono celular. El algoritmo de Viola-Jones traduce las características de la imagen y el procesador Jetson Nano hace un diagnóstico de coincidencia de características.

Figura 11

Fotografía de una persona con característica de pelo negro

Nota. Niño de 10 años con pelo negro.

La figura 12 compara la imagen de un rostro facial con lentes, obtenida a través de la cámara web en tiempo real, y hace un reconocimiento de los lentes como una característica resaltante.

Figura 12

Comparación de imagen obtenida de una persona con lentes a través de la cámara web en tiempo real y reconocimiento de lentes

Nota. Persona adulta con lentes.

Las siguientes fotografías son procesadas a través de la cámara web, frente a la persona en tiempo real. La figura 13 compara la imagen obtenida a través de la cámara web en tiempo real y hace un reconocimiento de los gestos de la persona enojada como una característica resaltante. La figura 14 compara la imagen obtenida a través de la cámara web en tiempo real y hace un reconocimiento de los gestos de la persona en estado de felicidad, como una nueva característica resaltante. Finalmente, en la figura 15 se compara la imagen obtenida a través de la cámara web en tiempo real y se realiza un reconocimiento de la persona, etiquetada con su nombre, de acuerdo con la base de datos proporcionada al sistema.

Figura 13

Comparación de la imagen obtenida a través de una cámara web en tiempo real y reconocimiento de gestos faciales de enojo

Nota. Persona adulta enojada.

Figura 14

Comparación de imagen obtenida a través de la cámara web en tiempo real y reconocimiento de gestos faciales de felicidad

Nota. Persona adulta feliz.

Figura 15

Comparación de imagen obtenida a través de la cámara web en tiempo real y reconocimiento del nombre completo de la persona y datos complementarios

Nota. Nombre de la persona.

CONCLUSIONES

- El algoritmo propuesto puede ser implementado en procesadores comerciales Jetson Nano NVIDIA con características de procesamiento bastante altos y con un bajo consumo energético.
- El algoritmo de Viola-Jones es fácil de implementar si se cuenta con una imagen en la tarjeta de memoria micro-SD que contenga todos los *frameworks* adecuadamente instalados.
- El lenguaje Python, combinado con los nuevos *frameworks* desarrollados y de libre instalación, hace que estos sistemas tengan un costo de programación aceptable y de rápido desarrollo; sin embargo, se necesita una buena calidad en la cámara web para que el algoritmo cumpla con los requerimientos mínimos de luminosidad.
- Si bien el algoritmo aún no trabaja de forma combinada con todas las opciones de identificación de parámetros, tanto gestuales como características de rostro con lentes, esta investigación abre una importante oportunidad para nuevos desarrollos en el campo del uso de la identificación de personas con ciertas características, por ejemplo, un tatuaje como señal distintiva, al igual que el reconocimiento facial.

- No solo la inteligencia artificial puede ayudar a predecir imágenes y dar como resultado una acción de alerta, sino que también puede combinar acciones tales como habilitar las salidas digitales o PWM de la tarjeta de desarrollo Jetson Nano. Un ejemplo es encender una luz de alerta en la caseta de los vigilantes en el Metropolitano o encender una alarma sonora para tener una acción preventiva, embebido en el procesador Jetson Nano.
- Si estos sistemas se encuentran, por ejemplo, en la línea de buses del Metropolitano de Lima, si se identifica a una persona que es buscada, se sabe en qué estación ha subido, en que vehículo se encuentra y en qué estación se ha retirado, reduciendo el espacio de búsqueda, y con ello se focaliza el accionar de la policía o autoridades de control.
- La inteligencia artificial no solo puede estar restringida al reconocimiento facial, sino también al reconocimiento del número de placa de los vehículos y modelos de automóviles, junto a sus propias características.
- En una línea de producción, con la inteligencia artificial se puede detectar características como una etiqueta mal puesta, la falta de una pastilla en un blíster o un componente electrónico. La inteligencia artificial puede ser aplicada a diferentes situaciones; no obstante, este algoritmo es bastante apropiado para el reconocimiento facial.

REFERENCIAS

Andrago Calvachi, M. A. (2019). *Uso de reconocimiento facial de emociones basado en técnicas de deep learning para el mejoramiento de la educación* [Tesis de maestría, Universidad Israel]. Universidad Israel, Repositorio Digital. http://repositorio.uisrael.edu.ec/handle/47000/2297

Duró, V. E. (2001). *Evaluación de sistemas de reconocimiento biométrico*. Escuela Universitaria Politécnica de Mataró, Departamento de Electrónica y Automática.

Estévez Martín, A., & Ramírez Barredo, B. (2018). Smartcity: la inteligencia artificial en la ciudad del futuro. Estudio del caso Amazon Go. En *Actas ICONO14. VI Congreso Internacional Ciudades Creativas* (pp. 199-215). Asociación de Comunicación y Nuevas Tecnologías.

Gualdrón, O. E., Duque Suárez, O. M., & Chacón Rojas, M. A. (2013). Diseño de un sistema de reconocimiento de rostros mediante la hibridación de técnicas de reconocimiento de patrones, visión artificial e IA, enfocado a la seguridad e interacción robótica social. *Mundo FESC*, *3*(6), 16-28.

Lienhart, R., & Maydt, J. (2002). An extended set of Haar-like features for rapid object detection. *Proceedings. International Conference on Image Processing* (vol. I, pp. 900-905). DOI: 10.1109/ICIP.2002.1038171

Planells Lerma, J. (2009). *Implementación del algoritmo de detección facial de Viola-Jones* [Trabajo de fin de carrera]. Universitat Politècnica de València. Escola Tècnica Superior d'Enginyeria Informàtica.

Viola, P., Jones, M. J., & Snow, D. (2005). Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, *63*(2), 153-161.

Wei, Y., Bing, X., & Chareonsak, C. (2004). FPGA implementation of AdaBoost algorithm for detection of face biometrics. En *IEEE International Workshop on Biomedical Circuits and Systems* (pp. S1/6-17). DOI: 10.1109/BIOCAS.2004.1454161

Zerrouki, N., Harrou, F., Sun, Y., & Houacine, A. (2018). Vision-based human action classification using adaptive boosting algorithm. *IEEE Sensors Journal*, *18*(12), 5115-5121. DOI: 10.1109/JSEN.2018.2830743