UNIVERSIDAD DEL AZUAY

FACULTAD DE CIENCIA Y TECNOLOGÍA

ESCUELA DE INGENIERÍA ELECTRÓNICA

“Desarrollo de un software como herramienta para el reconocimiento de patrones faciales caninos”

Trabajo de graduación previo a la obtención del título de:
INGENIERO ELECTRÓNICO

Autores:
Carlos Emilio Burbano López
Marcos Leandro Castro Ordóñez

Director:
MSc. Francisco Salgado

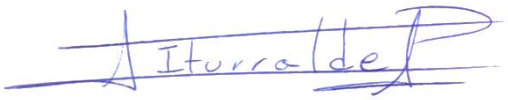
CUENCA-ECUADOR
2021

Desarrollo de un software como herramienta para el reconocimiento de patrones faciales caninos

RESUMEN

El siguiente estudio presenta el desarrollo de un software que tiene como objetivo reconocer de forma automática la identidad de un can mediante una fotografía, evaluando los patrones faciales por medio del procesamiento de imágenes e inteligencia artificial, ya que a una persona se le dificulta reconocer los rasgos físicos de un canino que observó en algún anuncio o en ciertos casos, este no posee otro método de identificación. El software fue desarrollado en Google Colab, donde se utilizó Inception-Resnet v2 para la detección facial, además la biblioteca Pytorch y las técnicas Support Vector Classifier y Directed Acyclic Graph Support Vector Machines, las mismas que sirven para la identificación canina, obteniendo como resultado una precisión promedio del 93,10% en el reconocimiento entre dos clases y del 90,30% entre cuatro clases.

Palabras clave—Directed Acyclic Graph, Google Colab, Incrustaciones, Inteligencia Artificial, Pytorch, Reconocimiento facial canino, Support Vector Machines, Machine-Learning, Deep-Learning.



Ing. Daniel Iturralde. PhD

Coordinador de Escuela

Ing. Francisco Salgado. Msc

Tutor de Tesis

Autores

Carlos Emilio Burbano López

Autor

Marcos Leandro Castro Ordóñez

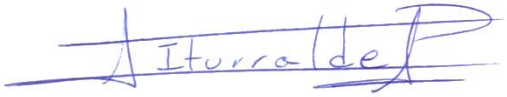
Autor

Development of software as a tool for canine facial pattern recognition

ABSTRACT

This study presents the development of a software to automatically recognize the identity of a dog in a photograph by evaluating facial patterns through images processing and artificial intelligence, since a person finds it difficult to recognize the physical features of a canine they have observed in an advertisement or in certain cases. Dogs don't have another method of identification. The software was developed in Google Colab. Inception-Resnet v2 was used for facial detection, in addition to the Pytorch library and the Support Vector Classifier and Directed Acyclic Graph Support Vector Machines techniques which are used for canine identification, obtaining as a result an average precision of 93.10% in the recognition between two classes and 90.30% between four classes.

Keywords: Directed Acyclic Graph, Google Colab, Embeddings, Artificial Intelligence, Pytorch, Canine Facial Recognition, Support Vector Machines, Machine-Learning, Deep-Learning.



Engr. Daniel Iturralde. PhD

School Coordinator

Engr. Francisco Salgado. Msc

Thesis Advisor

Authors

Carlos Emilio Burbano López

Author

Marcos Leandro Castro Ordóñez

Author

Translated by

Carlos Burbano and Marcos Leandro Castro

Desarrollo de un software como herramienta para el reconocimiento de patrones faciales caninos

Emilio Carlos Burbano López
Escuela de Ingeniería Electrónica
Universidad del Azuay, UDA
Cuenca, Ecuador
emilio1195@es.uazuay.edu.ec

Marcos Leandro Castro Ordóñez
Escuela de Ingeniería Electrónica
Universidad del Azuay, UDA
Cuenca, Ecuador
marcosleandrocastro@es.uazuay.edu.ec

Resumen*—Las personas que tienen una mascota conocen el valor sentimental que dicho animal otorga y lo importante que es para la vida del dueño. El siguiente estudio presenta el desarrollo de un software para reconocer de forma automática la identidad de un can, mediante una fotografía que puede ser tomada con cualquier teléfono celular, para facilitar el reconocimiento de identidad del perro extraviado, evaluando los patrones faciales, por medio del procesamiento de imágenes e inteligencia artificial, ya que a una persona se le dificulta reconocer los rasgos físicos del canino que observó en algún anuncio o en ciertos casos no posee otro método de identificación. El software fue desarrollado en Google Colab, donde se utilizó Inception-Resnet v2 para la detección facial, además la biblioteca Pytorch y las técnicas Support Vector Classifier y Directed Acyclic Graph Support Vector Machines sirven para la identificación canina, obteniendo como resultado en el reconocimiento entre dos clases una precisión promedio del 93,10% y entre cuatro clases el 90,30%, siendo el método más eficiente Directed Acyclic Graph Support Vector Machines, para trabajar con un dataset de entrenamiento pequeño. Los resultados de este estudio se pueden mejorar a futuro implementando una técnica de pérdidas de tripletes u otros métodos más sofisticados.

Palabras clave*—Directed Acyclic Graph, Google Colab, Incrustaciones, Inteligencia Artificial, Pytorch, Reconocimiento facial canino, Support Vector Machines, Machine-Learning, Deep-Learning.

I. INTRODUCCIÓN

La tecnología ha evolucionado tanto en el área investigativa como productiva y ha entregado herramientas para el desarrollo de áreas que no han sido explotadas siendo una de ellas, el campo de mascotas extraviadas; según datos de la fundación ARCA en Cuenca existen alrededor de 20.000 canes que han sido abandonados o extraviados [1]. El método más utilizado es el chip de identificación, el cual se inserta en áreas específicas (en la zona dorsal del cuello o detrás de la oreja izquierda) mediante un proceso invasivo. Este procedimiento presenta un funcionamiento limitado que, a diferencia de un rastreador satelital, se activa únicamente cuando se encuentra dentro del campo de radiofrecuencia de un escáner determinado [2], es decir, el chip para perros ayuda únicamente si el can es llevado a una veterinaria que cuente con un escáner especial para entregar el código perteneciente a la mascota. Sus principales desventajas son: el alcance mínimo de detección es un método invasivo y para su detección únicamente se realiza en lugares autorizados o que tengan el escáner de lectura, por lo tanto, es poco convencional. Es importante tomar en cuenta que "El chip no es un GPS, no permite rastrear al animal en tiempo real" [3], en otras palabras, no entrega las coordenadas exactas donde se encuentre el can. Por otro lado, existen dispositivos de rastreo que son escasos en el mercado ecuatoriano por su baja demanda y alto costo, siendo uno de ellos el receptor GPS para mascotas.

Existen diversas formas para poder encontrar a un can extraviado, por ejemplo, publicarlo en redes sociales. Esto tiene la desventaja de que la gran parte de los usuarios podría ignorar la publicación, no lograrían distinguir de manera exacta al perro o, simplemente, no se llega a visualizar dicha publicación. Otro procedimiento es la colocación de carteles que resultan poco legibles o con falta de información.

Actualmente, se han creado distintas aplicaciones con la ayuda de la inteligencia artificial las cuales facilitan el desarrollo de una gran variedad de actividades, incluyendo a aquellas de la vida diaria. En el Ecuador no se ha presentado un desarrollo tecnológico notable en el campo del cuidado de las mascotas. Por esto, resulta factible crear un software que, mediante el procesamiento de imágenes e inteligencia artificial permita reconocer la identidad de un can, diferenciando la similitud física entre perros de una misma raza. Brindando así una herramienta útil para los usuarios, con la finalidad de reducir el porcentaje de mascotas extraviadas que no se pueden reencontrar con sus dueños. De esta forma, se podría innovar en tecnologías informáticas de áreas poco convencionales como la tratada en este trabajo.

A. Estado del arte

El desarrollo del trabajo se fundamenta en distintos artículos científicos que se basan en el reconocimiento facial utilizando herramientas de inteligencia artificial o mecanismos que ayudan en la detección de patrones característicos.

En la Universidad Nacional de La Plata, utilizaron códigos bidimensionales y posicionamiento para el reencuentro de mascotas con sus dueños, el cual describe un

modelo de solución que incluye el uso de un código QR, un GPS y un Smartphone que permita escanear el código QR para obtener los datos principales del dueño de la mascota [4].

En EE. UU., se realizó un trabajo de una aplicación móvil para encontrar a las mascotas perdidas, donde se abarca una base de datos que ayuda a la búsqueda de los animales registrados y de esa manera localizar al dueño [5].

Un estudio realizado por Lai, Xinyuan Tu y Yanushkevich propone una identificación de perros mediante biometría suave y redes neuronales, donde se aplican modelos de aprendizajes para poder determinar la identidad de una mascota por medio de fotografías. Se utilizaron rasgos biométricos "suaves" como la raza, altura o sexo; así como rasgos biométricos "duros" como las fotografías de los animales a identificar, dando como resultados una predicción del 90,8% y 91,29% [6].

En la Universidad de Novi Sad en Serbia, se desarrolló un proyecto enfocado en una solución para reconocer la identidad de perros usando inteligencia artificial, logrando así una precisión del 94,59% en la identificación de las mascotas. El trabajo está realizado en las arquitecturas Inception v3 y SSD Inception v2, se integró esta herramienta a una red social Pet2Net para poder realizar los distintos testeos [7].

En Vancouver, Canadá, se desarrolló un sistema para alertar al dueño de un perro perdido. El proyecto cuenta con una base de datos que en donde se crea un perfil con los datos del dueño para poder contactarlo. El sistema añade al can en una lista de animales perdidos y permite mandar notificaciones para poder alertar a la gente, realiza una búsqueda en una base de datos y encuentra el perfil de referencia [8].

En la ciudad de Los Ángeles, se desarrolló un dispositivo de mano para escanear el hocico del animal. El dispositivo encaja y, por medio de un arreglo de cámaras, realiza el escaneo de los patrones característicos del hocico para poder determinar la identidad; información utilizada para el transporte de animales, prevenir robos, identificar mascotas perdidas, entre otros [9].

En la Universidad George Washington, se creó LemurFaceID, el cual es un programa que permite reconocer la identidad de cada lémur sin importar la raza. Se realiza mediante reconocimiento facial el cual se fija en los patrones más relevantes que presenta cada animal. Han llegado a obtener una precisión del 97% en sus pruebas [10].

Bassaure, Cárdenas y Hernández [11], implementaron un sistema de seguridad para el ingreso de mascotas a sus hogares, este método funciona solo en canes. El sistema utiliza la inteligencia artificial, que por medio de redes neuronales realiza el reconocimiento facial detectando los patrones más característicos del can, donde estos son obtenidos mediante procesamiento de imágenes con la ayuda de la visión artificial.

Mougeot, Li y Jia [12], realizaron un estudio en el cual evalúan el aprendizaje profundo de la Inteligencia Artificial para la verificación y el reconocimiento del rostro canino. Al realizar los entrenamientos del dataset y pruebas para un grupo abierto de 48 perros diferentes, alcanzaron una precisión del 92% en la verificación y una precisión del 88% en el reconocimiento facial en un solo intento.

Emma M. Baxter, junto con su equipo de trabajo, desarrollaron un sistema de reconocimiento facial en cerdos, con el fin de dejar a un lado el etiquetado tradicional mediante RFID (identificación por radiofrecuencia) que resulta invasivo. Se han realizado pruebas mediante las tres técnicas adoptadas del reconocimiento facial humano: Fisherfaces, VGG-Face y su propio modelo de CNN entrenado mediante un conjunto de datos aumentados artificialmente. Se han obtenido tasas de precisión del 96,7% en 1553 imágenes [13].

En la ciudad de Cha-An en la provincia China de Anhui, se ha desarrollado una aplicación llamada GoGo Chicken, que utiliza el reconocimiento facial para distinguir a las gallias, con el fin de que los consumidores puedan comprobar fácilmente el lugar de nacimiento, la alimentación y la información sanitaria del animal recién comprado, sin tener que recurrir a etiquetas o chips, además la implementación del internet de las cosas (IoT) en dispositivos de tobilleras que envían información vital y de actividad realizada del ave [14].

La implementación que se presenta en este trabajo se diferencia sede otras soluciones en que, principalmente, se hace uso de un entorno web de programación: Google Colab. Es decir, todo se encuentra y se ejecuta en la nube y no requiere instalación alguna. La mayor ventaja es que Colab facilita el uso de paquetes de aprendizaje automático (Machine Learning) más populares del lenguaje de programación Python que se pueden cargar fácilmente en el documento de desarrollo de código. Además, se utilizan tres métodos de reconocimiento de identidad mediante incrustaciones de los rasgos faciales caninos. Se realizó un entrenamiento de los modelos con el menor número de fotos obteniendo una predicción con resultados prometedores, también se normalizó las imágenes para mantener un equilibrio entre los datos de entrenamiento y de evaluación y el modelo de detección facial canina fue entrenado mediante un dataset de nuestra autoría.

B. Marco Teórico

a) Herramientas para la realización del software

Google Colab

Es un entorno de programación en línea basado en el estilo de Jupyter Notebook, al cual se accede desde un navegador con servicio de internet. Su finalidad es utilizar y mantener todos los archivos que se guardan en la nube [15].

Este servicio de Google ha mejorado con la proyección de Open AI, para tener la ayuda de muchos desarrolladores e ir mejorando las técnicas de programación en la Inteligencia Artificial, con el objetivo de que sea accesible para todas las personas que usen el entorno. Para que los usuarios de Gmail utilicen este servicio se han incorporado GPU-s (graphics processing unit) en sus computadoras con 12GB de RAM. Se accede a ellas mediante un enlace virtual, así los usuarios no dependen de un computador con limitaciones de memoria al ejecutar un código o hacer entrenamientos con Machine Learning [15].

Por otro lado, se debe tener en cuenta las desventajas que presenta el uso gratuito de este entorno. Este servicio tiene una alta demanda y se reinicia cada 12 horas; al apagarse la máquina virtual los archivos y librerías cargadas se eliminan, por tanto, es necesario tener un respaldo. Esto se puede mejorar con Google Colab Pro, el cual permite acceder al doble de recursos informáticos de lo que ofrece su forma gratuita [15].

Visión Artificial

También llamada visión por computador, es una técnica de inteligencia artificial implementada en Deep Learning o aprendizaje profundo para la construcción y entrenamiento de redes neuronales convolucionales, permitiendo de este modo detectar y descifrar patrones [16]. La visión artificial permite entrenar modelos y compilarlos de manera que se puedan depurar y mejorar, esta técnica necesita de entrenamiento que le ayudan a realizar el algoritmo para una detección con una mayor probabilidad de predicción, estos se adaptan para cualquier tipo de recurso informático, pero influye en su precisión a la hora de detectar. Debido a esto la mejor plataforma para este objetivo en tiempo real es Faster R-CNN Inception v2, el cual está optimizado para la detección facial [17].

Ilustración 1 Representación gráfica de un píxel [16].

Redes Neuronales Artificiales (RNAs)

Las redes neuronales artificiales buscan "emular el comportamiento del cerebro humano, caracterizado por el aprendizaje a través de experiencias y extracción de conocimiento genérico a partir de conjunto de datos" [18]. Las máquinas pueden aprender por sí solas mediante error y experiencia con el objetivo de hacer predicciones utilizando la información otorgada.

Existen varios métodos que involucran una red neuronal, por su principio de funcionamiento y su estructura mediante grafos que están conectados entre sí. El funcionamiento se basa en obtener información de entrada en los nodos principales que están conectados a otros nodos que conforman la capa oculta. En esta capa trabajan los nodos internamente realizando diferentes funciones, toman una primera muestra para realizar una predicción o clasificación, la cual será enviada a la segunda capa oculta donde tomarán otra muestra y así sucesivamente. Se realiza esto con el objetivo de ir consumiendo la información de entrada dependiendo de la tarea que se quiere realizar para obtener una salida [18].

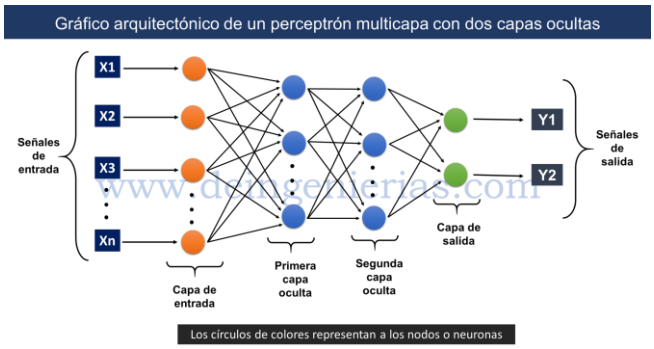


Ilustración 2 Estructura de una Red Neuronal Artificial [18].

Pytorch

Pythorch es una librería de tensores optimizada para aprendizaje profundo (Deep Learning) basada en la biblioteca Torch, utilizada en aplicaciones de visión artificial y procesamiento del lenguaje natural. Sus principales características son los cálculos de tensores con un fuerte soporte de aceleración de GPU y la construcción de redes neuronales profundas en sistemas de autogrado [19], además una gran ventaja es su diseño modular y flexibilidad por lo que, no se limita a aplicaciones específicas.

Un tensor es un término que se utiliza como una estructura de datos en Deep Learning, es decir, es un término genérico de matriz o vector, por lo cual un tensor de rango uno es equivalente a un vector y así sucesivamente.

Máquinas de vectores de soporte (SVM)

El support vector machine es un algoritmo de aprendizaje automático supervisado [20], el cual es utilizado para problemas de clasificación. Dicho algoritmo encuentra una cantidad n de números de características que posee el elemento a analizar y este posee una coordenada en particular. Una gran ventaja que tiene este algoritmo es que se puede usar un kernel (también llamado núcleo), el cual ayuda a convertir un problema no separable a uno separable, ayudando en desafíos de separación no lineal [20].

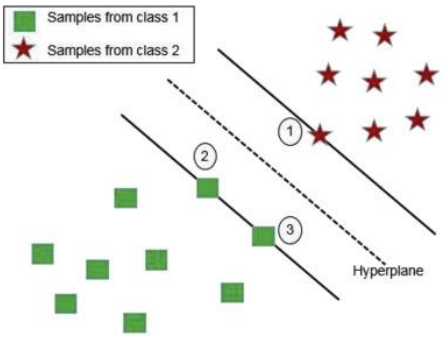


Ilustración 3 Separación del hiperplano con SVM [20].

Gráfico Acíclico Dirigido SVM (DAGSVM)

El objetivo de este es reducir el tiempo de ejecución en la clasificación. Para lograr esto, se elimina comparaciones que no son necesarias con la ayuda de un gráfico acíclico dirigido, DAG, por sus siglas en inglés. Dirigido hace referencia que

cada borde tiene una dirección definida, representando un flujo direccional único de un vértice a otro. Acíclico significa que no existen bucles, por lo que para cualquier vértice no hay camino en el gráfico para volver a ese vértice inicial [21]. Por ejemplo, en un grupo de datos con cuatro clases (1, 2, 3, 4) y se tienen 6 clasificadores entrenados con cada par posible (1-2, 1-3, 1-4, 2-3, 2-4, 3-4). El clasificador se inicia con el par (1-4) y da como resultado que la pertenece a la clase 1. Con esta conclusión, se eliminan todas las comparaciones que contengan la clase 4, ya que se conoce que no pertenece a este grupo. La siguiente paridad entre (1-3) el algoritmo predice que pertenece a 1, del mismo modo se elimina las comparaciones que contengan a 3 y por último el par (1-2) y se sigue la tendencia. De este modo, se reduce el tiempo de cómputo al no tener que evaluar los 6 emparejamientos sino únicamente 3 para poder realizar la clasificación [22].

Biometría

La Biometría consiste en medir e identificar características biológicas, siendo las más comunes en los humanos: la huella digital, el rostro y la voz, ya que han sido utilizadas para la identificación de una persona. "Pero los humanos no somos los únicos que tenemos rasgos característicos, los animales cuentan con ellos también" [23]. De manera que se ha implementado tecnología que abarque la biometría en animales, con el objetivo de evitar el mercado ilegal de especies y el reconocimiento de mascotas extraviadas.

b) Anatomía canina

Nariz canina

La nariz de un can contiene datos característicos, los cuales ayudan a diferenciar un perro de otro, ya que su biometría puede variar en la forma de la nariz, tamaño del contorno, tamaño de los orificios respiratorios, distancias de extremos al punto central, longitud del surco, entre otras [24]. "No es la primera vez que se usa la nariz de los perros como método de identificación, la novedad está en el método y los resultados" [25].

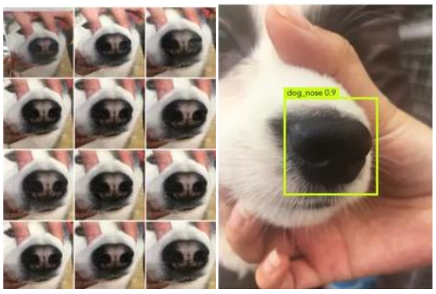


Ilustración 4 Data Set nariz canina, detectando nariz [25].

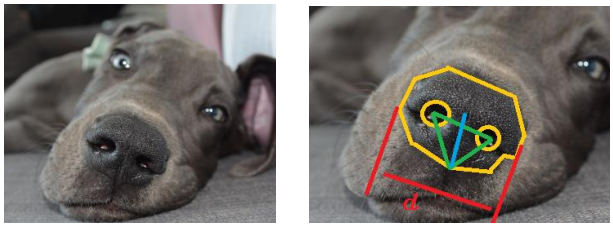


Ilustración 5 Ejemplo de posibles datos biométricos de la nariz de un can [25].

Regiones naturales y referencias de superficie de la cabeza

Esta zona engloba a cada región que tiene una forma característica, siendo una de ellas la forma de la cabeza, donde se encuentran los ojos, el hocico y las orejas [24]. Se puede extraer de estas zonas el tamaño de la cabeza, tamaño de hocico, distancias de extremos, tomándolos desde una vista lateral del can.

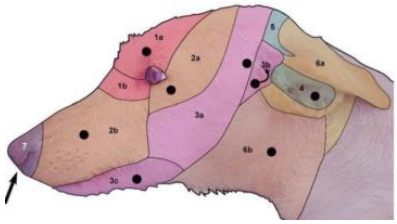


Ilustración 6 Vista lateral de la cabeza del can con sus regiones características [24].

Por otro lado, se puede identificar el contorno de la cabeza mediante una vista frontal, en dicha posición se puede apreciar la postura de las orejas para obtener un contorno total y conocer la distancia que existe entre sí.

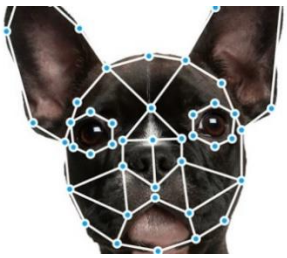


Ilustración 7 Biometría de la vista frontal de la cabeza del can [26].

Ojo

La biometría del ojo tiene un papel muy importante en los canes ya que poseen iris, el cual puede ser detectado para ser procesado obteniendo su contorno y sus dimensiones. La detección del iris de un animal puede ser tratada de forma similar a la detección del de un ojo humano [27].

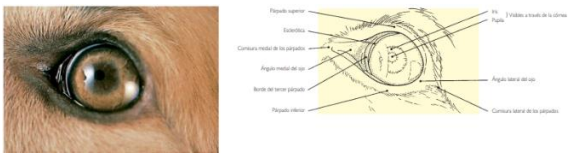


Ilustración 8 Anatomía ojo de un can [27].

II. METODOLOGÍA

A. *Introducción*

La implementación del software para el reconocimiento de patrones faciales caninos está compuesta por programas independientes como detectores, entrenamientos, testeos y un almacenamiento en la nube, los cuales ayudan a la realización del trabajo y encontrar la forma más eficaz en obtener los resultados, se realizaron distintos métodos con el fin de encontrar la mejor opción para un mejor reconocimiento.

B. *Conjunto de datos*

El conjunto de datos o dataset que se utilizó para la resolución del proyecto fue generado mediante un etiquetado

de forma manual en el programa LabelImg el cual es una herramienta de anotación de imágenes [28]. Se crearon tres etiquetas: *eye_dog*, *face_dog*, y *nose_dog*, con el fin de enfocarse únicamente en la cabeza del can. Se colocaron las etiquetas en las partes correspondientes de cada fotografía, las imágenes fueron obtenidas de fuentes disponibles en la web, las mismas que se descargaron con distinta calidad, diferentes razas de perros y diversos ángulos de su rostro, de este modo el dataset sea robusto y variado.

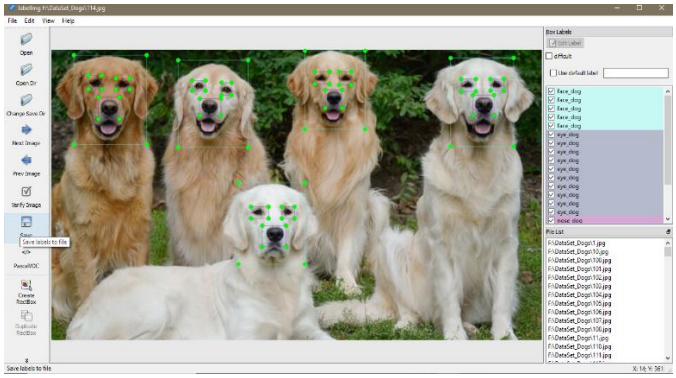


Ilustración 9 Etiquetado en el programa LabelImg.

Las imágenes fueron guardadas con el etiquetado correspondiente y se crea un archivo xml, el cual guarda la información de los límites de los cuadros de las respectivas etiquetas en el directorio seleccionado. Este tipo de formato es legible para el entrenamiento del modelo, en la detección de las regiones de interés.

Para ordenar los datos, estos fueron nombrados de forma secuencial numérica; en total, resultaron 614 imágenes que conforman el respectivo dataset, siendo dividido en dos partes: 515 fotografías para realizar el entrenamiento y 99 para evaluar el modelo.

C. Entorno de programación

El programa para el reconocimiento de patrones faciales caninos se realizó en el lenguaje de programación Python en el entorno de desarrollo Google Colab, el cual es un servicio en la nube que permite el uso gratuito de GPU. Se utilizó este entorno debido a que brinda los recursos computacionales necesarios para realizar el entrenamiento del software, además que facilita en la instalación de librerías para trabajar con inteligencia artificial.

En el entorno se encuentra el software con las herramientas utilizadas para desarrollarlo; el entrenador de detección de caras, técnicas de detección de patrones, región de interés (RoI), Support Vector Classifier (SVC), Pytorch [19] y Directed Acyclic Graph Support Vector Machines (DAGSVM).

D. Software de entrenamiento de detección facial canina

En el entrenamiento se utilizaron dos modelos para realizar las pruebas respectivas, el modelo ssd Resnet50 v1 fpn coco, por lo general este modelo se utiliza para la detección de objetos basados en ResNet; y el modelo Faster RCNN inception v2 pets, que está especializado en un conjunto de datos de mascotas, siendo utilizado por lo general para la detección de gatos, perros y reconocimiento de razas; este modelo da un mejor resultado como se observa en la Tabla I en la detección de las regiones de interés.

El *step* o paso contiene el número total de iteraciones que se usan en un entrenamiento, donde cada paso calcula la pérdida de clasificación que se produce en un lote y usa dicho valor para modificar los pesos del modelo [29].

El valor de pérdida en la clasificación es obtenido al aplicar la función de pérdida (*loss function*) en función de la probabilidad estimada del modelo. En el caso del primero, Se utiliza la función *focal loss* (FL) [30]:

$$FL(p_t) = -(1-p_t)^{\gamma}\log(p_t) \qquad (1)$$

Donde $(1-p_t)^{\gamma}$ es un factor de modulación a la pérdida de entropía cruzada donde γ se prueba a partir de [0,5] en el experimento.

El segundo modelo utiliza la función de pérdida descrita en [31] en detección de objetos:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*) \qquad (2)$$

El primer término es la pérdida de clasificación en 2 clases (hay objeto o no). El segundo término es la pérdida de regresión de los cuadros delimitadores (bounding box) solo cuando hay un objeto (es decir, $p_i^* = 1$).

TABLA I. COMPARACIÓN DE MODELOS ENTRENADOS.

Modelo	Step	Pérdida
Ssd resnet50 v1 fpn coco	114200	0.317
Faster RCNN inception v2 pets	149900	0.103

El modelo se puede seguir entrenando hasta los steps predefinidos, sin embargo, cuando la pérdida no baja de un cierto valor o se queda oscilando en un rango repetitivo, por lo que se recomiendo no seguir entrenando al modelo, debido a que no se observarán cambios y se gastarán recursos computacionales. Se utilizaron los modelos congelados para trabajar con los datos obtenidos en el último checkpoint creado y evaluar los resultados obtenidos por cada modelo. Según se observa en la Ilustración 10 reconoce todas las etiquetas entrenadas.

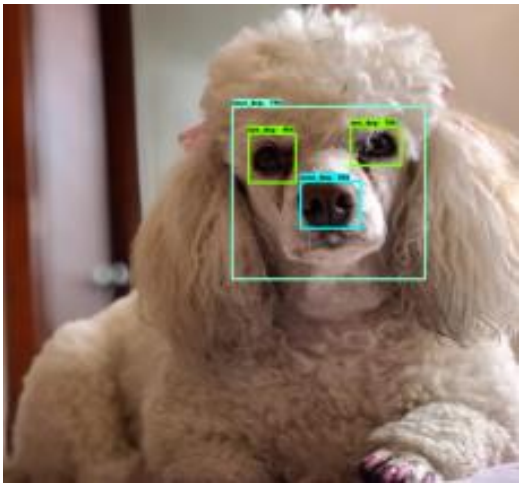


Ilustración 10 Resultado del modelo ssd Resnet50 v1 fpn coco.

E. Técnicas de detección de patrones

Se realizó un script que ayude a reconocer puntos característicos que tenga la imagen a procesar, para esto se utilizó *SIFT* y *ORB*, que son algoritmos de detección de patrones característicos en una imagen. El algoritmo *SIFT* realiza la transformación de las características invariantes de escala, es decir, se mantiene invariable a la rotación, al reescalado y los cambios de brillo que tenga una imagen. Una de las grandes ventajas es que es una función local muy estable.

Ilustración 11 Algoritmo Sift.

Oriented FAST and Rotated BRIE o simplemente ORB por sus siglas en inglés, usa supresión no máxima para mantener los puntos que existen en una esquina y de este modo predecir que en ese pixel se encuentra un punto característico [29]. Una ventaja de ORB es que puede especificar el número de puntos de vértice que se extraerán para dar un mejor resultado como se observa en la Ilustración 12.

Ilustración 12 Algoritmo ORB.

Como se observa en la Ilustración 11, el programa detecta puntos característicos en la grada de dicha fotografía; además, este no realiza correctamente el emparejamiento o *match* entre los canes, dando un margen de error elevado debido a que las fotos que tenían un cambio de contraste notable poseían los puntos característicos en diferentes posiciones. Como se aprecia en la Ilustración 13, existe un *match* debido a que no existen otras perturbaciones, por otro lado, como se evidencia en la Ilustración 14, si existen otras fotografías el algoritmo hace un emparejamiento con otras fotos que no pertenecen al mismo perro, además, si no se mantiene un ambiente controlado de la fotografía pueden variar las coordenadas de cada punto característico.

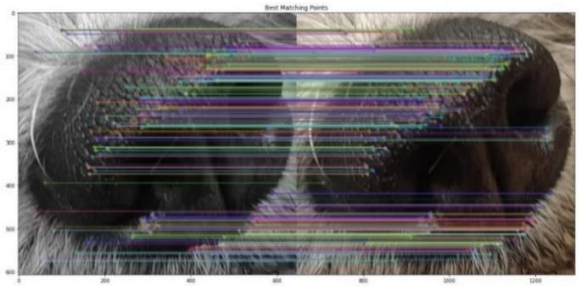


Ilustración 13 Match de imágenes similares.

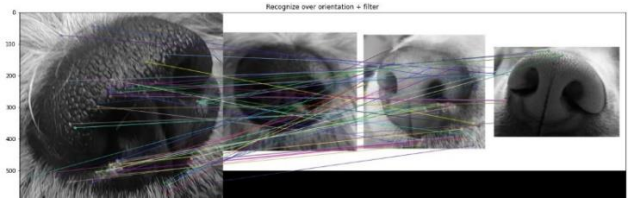


Ilustración 14 Error en el emparejamiento.

F. Detector de región de interés ROI

Con el modelo entrenado se realiza un script que permita encontrar la distancia que existe entre las regiones de interés de la imagen a evaluar. Primeramente, se reconoce en qué parte de la fotografía se encuentra la cara del can, para poder obtener el centroide del *RoI* etiquetado como *face_dog*.

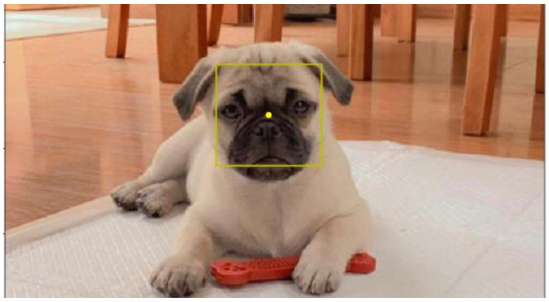


Ilustración 15 Obtención del centroide de la etiqueta face_dog.

Cuando se obtiene la detección de la etiqueta *face_dog*, se realiza la extracción de las demás etiquetas del entrenamiento. Cuando se obtienen los RoI, se calcula el centroide de cada sección de interés con el objetivo de calcular la distancia que existe entre los ojos y la nariz con el fin de determinar la identidad del can en base a los valores obtenidos.

Ilustración 16 Obtención de las etiquetas face_dog, nose_dog y eye_dog.

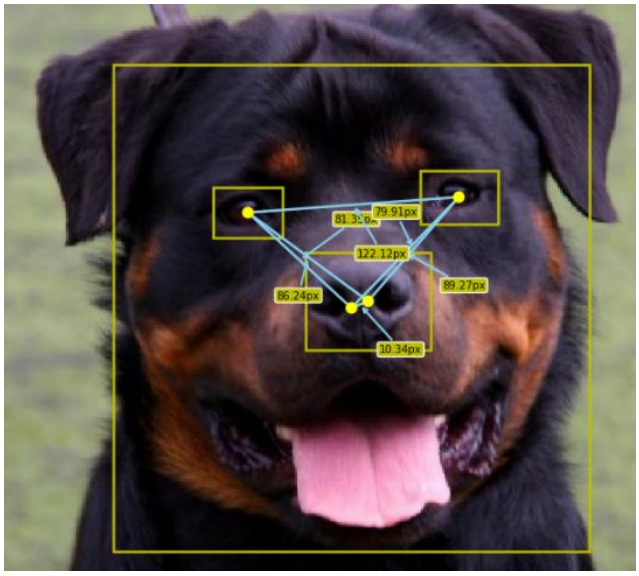


Ilustración 17 Calculo de las distancias entre las etiquetas.

Una inconveniencia de este método es que depende de la forma que es tomada la fotografía, debido a que las medidas entre las etiquetas varían dependiendo de la distancia con la que fue capturado el retrato del can. Además, el centroide del recuadro de la región de interés no se ubicaba en el centro real, es decir, el centroide de la etiqueta *eye_dog* no se ponía exactamente en el iris del ojo, causando una variación entre las distancias originales de las facciones del perro.

Se creó una máscara que permita reconocer la circunferencia del ojo y de este modo se pueda obtener las medidas correctas. Para ello se utilizó el ruido Gaussiano y la función *HoughCircles* que permite fijarse en regiones que sean una circunferencia.

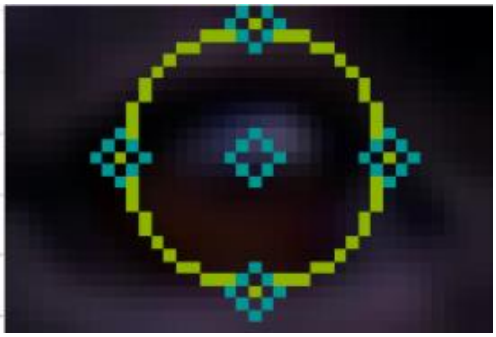


Ilustración 18 Función HoughCircles para detectar ojos.

HoughCircles depende de valores ingresados manualmente y dichos parámetros varían dependiendo de la distancia de la foto o del tamaño biométrico del ojo del can. Esto ocasiona que, al no mantener el mismo valor configurado en los parámetros de la función, esto provoca que exista un desfase del círculo detectado con respecto al ojo. Por lo tanto, las medidas obtenidas variarán dependiendo del tamaño del can y la distancia con la que haya sido tomada la fotografía, provocando que el resultado no sea óptimo, debido a su dificultad de calibración dicho método fue descartado.

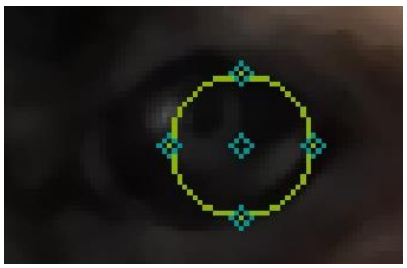


Ilustración 19 Desfase de la máscara con respecto al ojo.

G. Software para el entrenamiento del reconocimiento de la identidad canina

Se investigaron nuevas alternativas para utilizarlas en el reconocimiento facial como por ejemplo *Pytorch*, *SVC y DAGSVM*, con el objetivo de mejorar los resultados y obtener un software confiable en el reconocimiento. El diagrama de estados para la implementación del código se representa en la Ilustración 20.

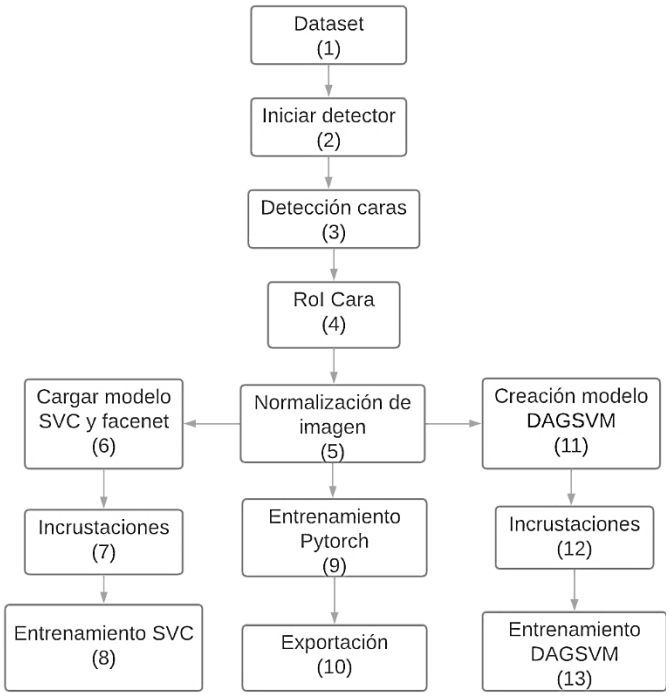


Ilustración 20 Diagrama de procesos del entrenador.

El bloque número uno realiza el ingreso del dataset que contiene las fotografías a entrenar, estas se encuentran organizadas por el nombre de cada can, en el bloque dos se carga el *model graph* obtenido mediante la red neuronal *Faster RCNN inception v2 pets,* para inicializar el detector de caras, una vez realizado lo anterior descrito se procede al bloque tres en donde se realiza la detección de caras de cada fotografía del dataset, por lo tanto, el bloque cuatro se encarga de extraer la región de interés con la ayuda de los limites obtenidos del bloque tres. Al tener la cara extraída almacenada en una variable, esta entra al bloque número cinco donde se encuentran las funciones de escalado y recorte, luego se carga el modelo SVC y *facenet* en el bloque número seis, para obtener las incrustaciones de la imagen normalizada, así proceder con el entrenamiento del modelo, de la misma manera se realiza el entrenamiento Pytorch mediante las incrustaciones y distancias euclidianas, las que serán almacenadas en una lista a exportar. Por último, el bloque once inicia la secuencia que crea el modelo DAGSVM donde se evalúa cada imagen para la obtención de las incrustaciones y obtener el modelo entrenado a exportar.

a) *Pytorch*

Se carga la fotografía del can para obtener los datos de sus incrustaciones y de este modo encontrar qué valores se asemejan más a las imágenes pre-entrenadas en el modelo.

Para entrenar dicha biblioteca de aprendizaje automático, se necesitaron 7 fotografías de cada can. Se utilizó el *dataset* creado al inicio, el cual permite fijarse únicamente en la cara del can y extraer los datos característicos de dicha región. El modelo ResNet que es utilizado por Pytorch necesita una imagen de entrada de 160x160 pixeles, lo cual se desarrolló una función para reescalar a dichas dimensiones.

b) Support vector classifier SVC

Al trabajar con *sklearn metrics* se obtiene el valor métrico de cada patrón de las capas ocultas de la red neuronal entregando como salida un vector de 1 x 128, por ello, para poder procesar estos datos y etiquetarlos con un respectivo *ID* se optó por utilizar SVC, la cual nos permite realizar un entrenamiento con el conjunto de datos del dataset de canes registrados, así poderlos posicionar en un plano de predicciones donde el algoritmo se encarga de organizar dichos datos separándolos mediante el hiperplano. Al momento de realizar la prueba se debe ingresar una fotografía para obtener la métrica, así el algoritmo entrega una predicción porcentual de la etiqueta correspondiente del can al que se asemeja más en la clasificación.

c) DAGSVM

Para la técnica de DAGSVM, se necesita obtener una lista de todas las combinaciones pares de los nombres de todos los canes registrados, se genera una máscara de todas las posiciones de las combinaciones, posteriormente realizar un filtrado de todas las ubicaciones y así guardar cada RoI del can con su ubicación respectiva en una nueva lista, luego se realiza el entrenamiento del clasificador para cada combinación con los datos de la lista anterior.

Para la predicción se utiliza las incrustaciones del can a encontrar para la prueba, los clasificadores, los nombres registrados y las combinaciones. En su interior se trabaja con un bucle donde se itera cada combinación para obtener la respectiva ubicación así obtener el clasificador correspondiente que predice mediante las incrustaciones encontradas; luego se ejecuta un método de descarte en el caso que no se encuentre la predicción con dicha combinación, esta será eliminada y se repetirá el bucle ignorando las clases que fueron descartadas, las cuales puedan encontrarse en alguna otra combinación y de este modo se evita realizar una predicción con una clase ya procesada anteriormente, hasta obtener una predicción.

d) Test de las herramientas

Cuando se obtienen los datos entrenados de cada herramienta, se realiza el testeo para obtener los resultados de cada algoritmo, para esto se carga una imagen que no haya sido vista por el programa, en donde cada herramienta realiza el cálculo de cada dato, como en el caso de Pytorch se obtienen las distancias euclidianas y las incrustaciones para poder encontrar un match con las imágenes cargadas en el entrenamiento, en SVC se adquieren las incrustaciones métricas de la fotografía ingresada y con los datos se encuentra el emparejamiento con las incrustaciones de las imágenes pre-entrenadas. El diagrama de bloques del test para el reconocimiento se encuentra en la Ilustración 21

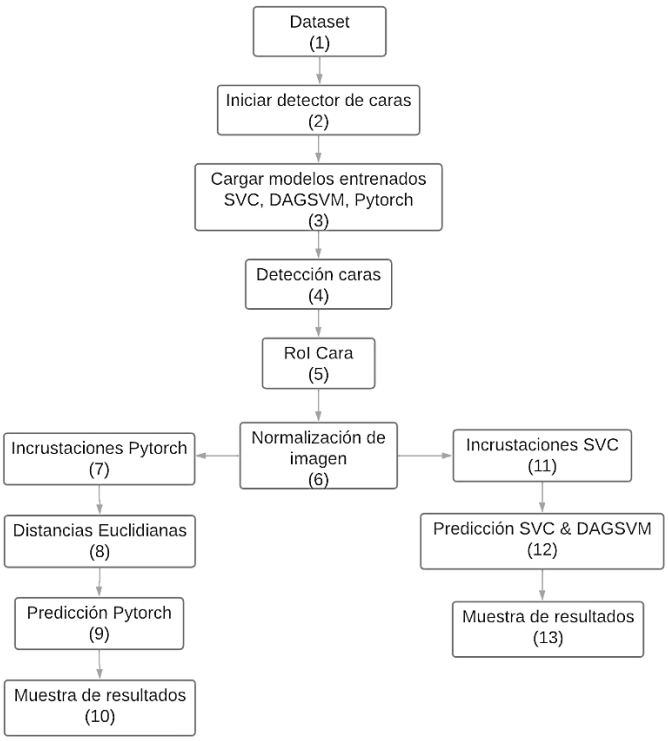


Ilustración 21 Diagrama de procesos de la prueba de reconocimiento.

Este diagrama, funciona de manera similar al diagrama de entrenamiento, con la diferencia de que el dataset de entrada son las fotografías de los canes a evaluar, además el bloque tres cargas los modelos entrenados para la predicción de la identidad del can y por último el bloque diez y trece muestran los resultados de predicción en una tabla de confusión y una tabla de pandas.

III. RESULTADOS

En base a los objetivos que se plantearon para la resolución del proyecto se realizaron las respectivas pruebas comparando los métodos Pytorch, SVC y DAGSVM, para un mejor entendimiento se realizó matrices de confusión, además se calculó la precisión, la sensibilidad y exactitud de la matriz que posee cada uno de los métodos propuestos, dichas métricas se calculan por medio de las siguientes fórmulas:

$$Precisión = \frac{TP}{TP + FP} \quad (3)$$

$$Exactitud = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

$$Sensibilidad = \frac{TP}{TP + FN} \quad (5)$$

Donde:

TP = Es el verdadero positivo.

TN = Es el verdadero negativo.

FP = Es el falso positivo.

FN = Es el falso negativo.

Como se observa en la Ilustración 22 se obtuvo el resultado del reconocimiento de identidad del can Beto, con respecto a los canes Bruno, Lita y Vale. Se realizó la prueba con los distintos métodos propuestos, el primer método Pytorch entrega con qué imagen del entrenamiento se realizó el match del reconocimiento.

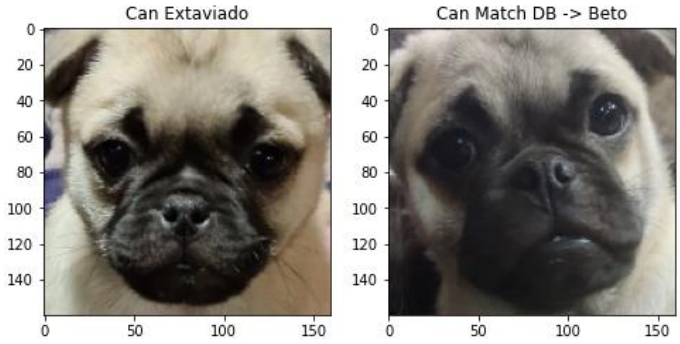


Ilustración 22 Resultado del match de Pytorch, con el can Beto.

Cabe recalcar que la imagen de la izquierda es la fotografía introducida al software y que no fue vista por el mismo, y la de la derecha es la fotografía con la que hizo match en el dataset de entrenamiento y prediciendo la identidad del can, en este caso Beto.

El método SVC y el de DAGSVM entregan los resultados del match con el can Beto, el resultado se adjunta en la Ilustración 23 respectivamente.

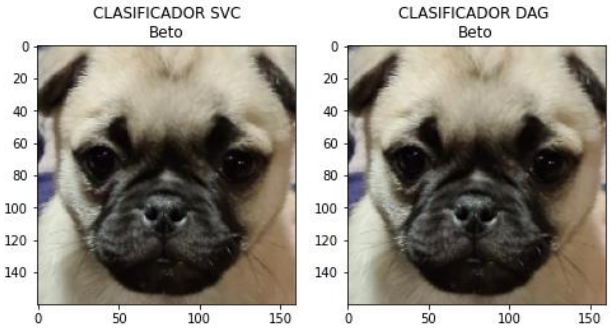


Ilustración 23 Resultado del clasificador SVC y DAGSVM.

Igualmente, se implementó una tabla con la librería Pandas, para observar el siguiente match que podría existir con Pytorch, para poder visualizar la primera y la segunda opción de los match posibles, con el fin de poder examinar las imágenes de los canes y distinguir cual es el perteneciente a la fotografía introducida en el programa.

Match del Can Extraviado con los siguientes canes

	Can	Distancia	Sexo
0	Beto	0.692839	Macho
1	Bruno	0.734817	Macho

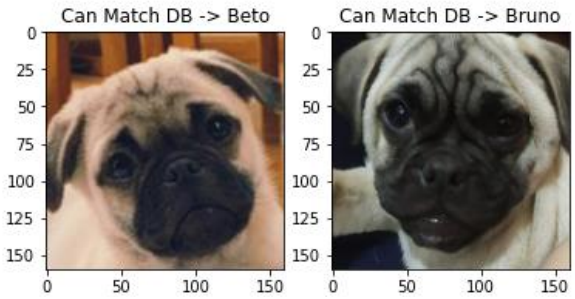


Ilustración 24 Visualización de Pandas por medio del método Pytorch, para la visualización del match y posible match.

Las Ilustraciones 25 y 26 visualizan los resultados del reconocimiento con el can Beto, del mismo modo se obtienen los resultados de los distintos métodos propuestos.

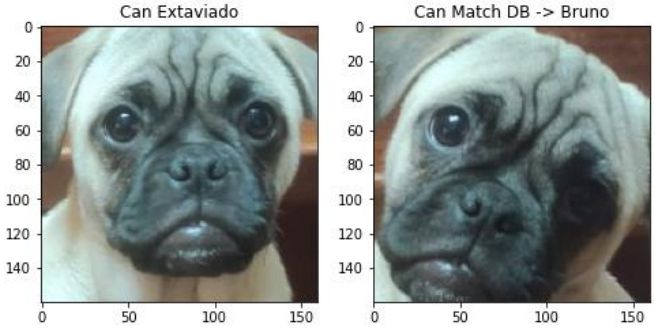


Ilustración 25 Resultado del método de Pytorch.

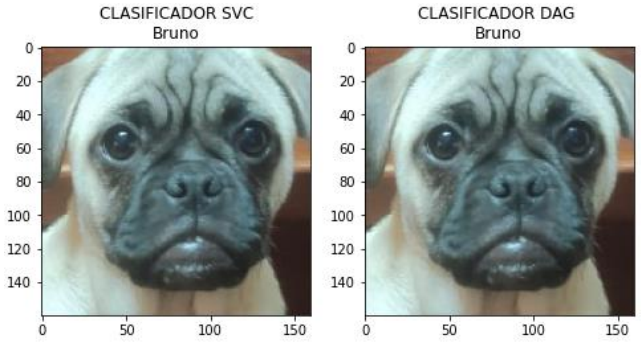


Ilustración 26 Resultado del método de SVC y DAGSVM.

Para tener un mayor conocimiento del rendimiento de los modelos, se evaluaron 25 imágenes de los canes Beto y Bruno. Se obtuvieron matrices de confusión para poder visualizar los testeos, además de los valores de precisión, exactitud y sensibilidad de cada matriz. se calcularon por medio de las siguientes fórmulas:

Los valores se obtienen de la matriz de confusión de cada método.

Ilustración 27 Canes de la misma raza para la prueba, Beto y Bruno respectivamente.

En la TABLA II se observan los resultados de precisión, exactitud y sensibilidad que tiene método Pytorch, y en la Ilustración 28 se observa la matriz de confusión del reconocimiento realizado con dos canes de la misma raza.

TABLA II. VALORES DE PRECISIÓN, EXACTITUD Y SENSIBILIDAD CON PYTORCH.

Can	Precisión	Exactitud	Sensibilidad
Beto	78.571	82.0	88.0
Bruno	86.364	82.0	76.0

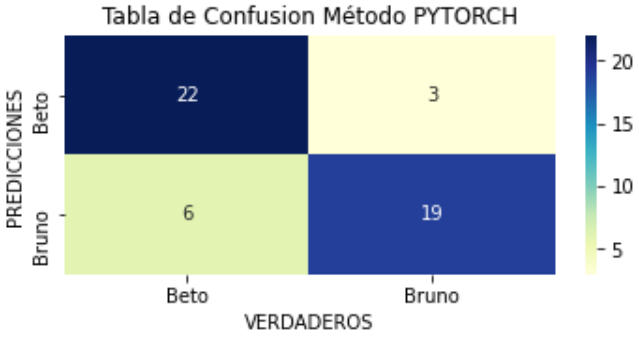


Ilustración 28 Matriz de confusión con Pytorch.

En la TABLA III, se aprecian los valores obtenidos en el método de SVC, del mismo modo la TABLA IV presenta los datos pertenecientes al método de DAGSVM. En las Ilustraciones 29 y 30 se muestra las matrices de confusión de los métodos SVC y DAGSVM respectivamente.

TABLA III. VALORES DE PRECISIÓN, EXACTITUD Y SENSIBILIDAD CON SVC.

Can	Precisión	Exactitud	Sensibilidad
Beto	80.769	82.0	84.0
Bruno	83.333	82.0	80.0

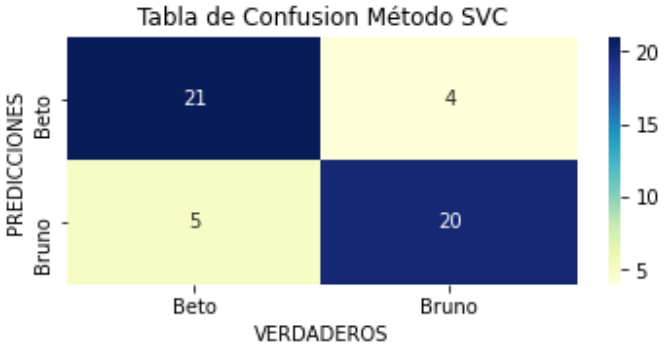


Ilustración 29 Matriz de confusión del con SVC.

TABLA IV. VALORES DE PRECISIÓN, EXACTITUD Y SENSIBILIDAD CON DAGSVM.

Can	Precisión	Exactitud	Sensibilidad
Beto	86.207	92.0	100.0
Bruno	100.0	92.0	84.0

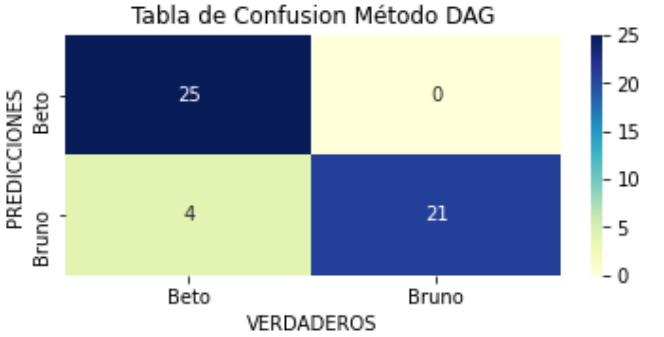


Ilustración 30 Matriz de confusión con DAGSVM.

Además, se realizó la prueba en un conjunto de cuatro canes para analizar los resultados del software cuando existe una mayor cantidad de información, los canes testeados son Beto, Bruno, Lita y Vale, los dos primeros perros son de raza Pug y los dos segundos Chihuahuas.

Los resultados obtenidos se muestran en las matrices de confusión de las Ilustraciones 31-33, realizado con los métodos de Pytorch, SVC y DAGSVM respectivamente. Los valores de precisión, exactitud y sensibilidad se encuentran en las TABLAS V-VII, con los valores de cada método anteriormente descritos.

TABLA V. Valores técnicos del con Pytorch, evaluado con 4 canes.

Can	Precisión	Exactitud	Sensibilidad
Beto	67.742	73.0	84.0
Bruno	77.273	73.0	68.0
Lita	75.0	73.0	72.0
Vale	73.913	73.0	68.0

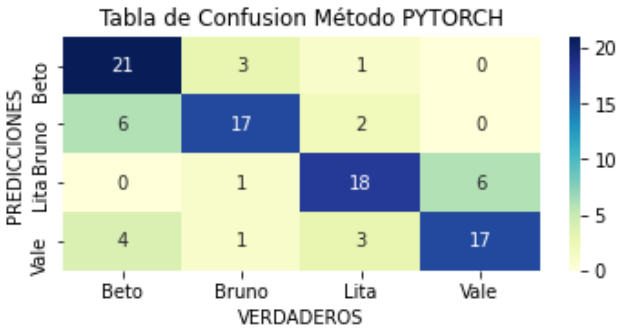


Ilustración 31 Matriz de confusión con Pytorch.

TABLA VI. VALORES DE PRECISIÓN, EXACTITUD Y SENSIBILIDAD CON SVC, EVALUADO CON 4 CANES.

Can	Precisión	Exactitud	Sensibilidad
Beto	84.0	77.0	84.0
Bruno	82.609	77.0	76.0
Lita	67.742	77.0	84.0
Vale	76.19	77.0	64.0

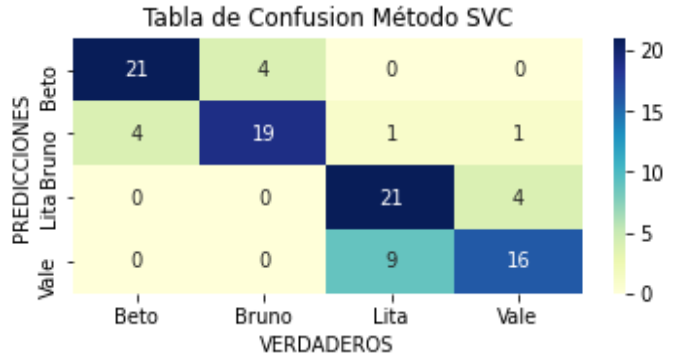


Ilustración 32 Matriz de confusión con SVC.

TABLA VII. VALORES DE PRECISIÓN, EXACTITUD Y SENSIBILIDAD CON DAGSVM, EVALUADO CON 4 CANES.

Can	Precisión	Exactitud	Sensibilidad
Beto	86.207	89.0	100.0
Bruno	100.0	89.0	84.0
Lita	80.0	89.0	96.0
Vale	95.0	89.0	76.0

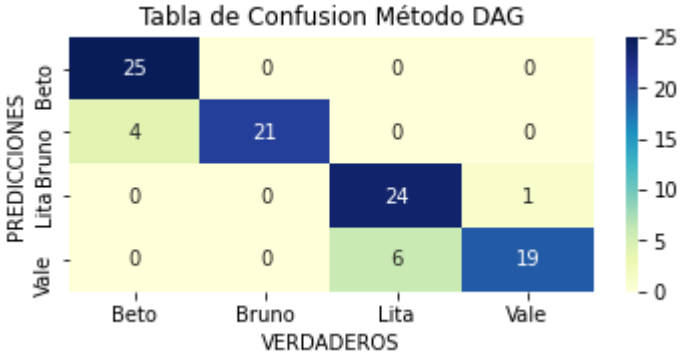


Ilustración 33 Matriz de confusión con DAGSVM.

IV. CONCLUSIONES Y RECOMENDACIONES

Para iniciar con el desarrollo del proyecto se partió con una investigación sobre los rasgos biométricos caninos, donde se trabajó con todos los rasgos faciales del can obteniendo mejores resultados. Ya que, al procesar un solo rasgo biométrico facial, las dimensiones de estos suelen ser menor a la dimensión de la imagen de entrada del modelo de incrustaciones (160x160 píxeles) dando perdidas de calidad y características al momento de escalarlas a una mayor dimensión.

Las pruebas realizadas mediante cada método presentado anteriormente para el reconocimiento facial canino presentan resultados prometedores al mantener un dataset pequeño para cuatro canes, en el cual se utilizó siete fotos de cada uno para el entrenamiento del modelo clasificador. Se realizó un entrenamiento del algoritmo con la cantidad mínima de fotos; así que, los entrenamientos y predicciones fueron acertadas correctamente, pero estos llegaban a tener consecuencias cuando se aumentaba el número de canes en el dataset disminuyendo el rendimiento de los clasificadores.

Al utilizar modelos de clasificación como SVM, árboles de decisión, random forest, entre otros, se debe manipular correctamente las matrices respectivas de las fotos de los canes, ya que cada modelo tiene diferente forma de procesarlos dependiendo el formato de los datos que se ingresarán, en este caso se ha utilizado SVC perteneciente a *support vector machines*, este se utiliza para la clasificación binaria (dos clases), sin embargo, este detecta automáticamente si se está trabajando con un mayor número de clases para organizarlos en el mismo espacio de trabajo, esto daría complicaciones en la predicción dando confusiones al momento de identificar correctamente al canino ingresado, por esta razón se utilizó un método multi-clase para tener un modelo general controlado y poder separar las clases, reduciendo la probabilidad de confusión y mejorando el resultado final con mayor velocidad gracias a la técnica empleada descrita en la teoría.

Mediante los resultados finales se ha llegado a concluir que el método más eficiente es DAGSVM, para trabajar con un dataset de entrenamiento pequeño y con modelos de incrustaciones que no fueron generadas específicamente para canes.

Para reducir errores de predicción se recomienda, utilizar la técnica de *triplet loss*, ya que esta da un ajuste único en el valor de las incrustaciones de cada can, permitiendo diferenciar rasgos similares del can a evaluar con respecto a otros canes; por otro lado, al querer obtener una mejor predicción con menor cantidad de fotos de entrenamiento, se puede utilizar OpenCV para generar imágenes alternas partiendo desde una foto mediante el uso de filtros, ya que las incrustaciones también se enfocan en las tonalidades y texturas de cada foto. Al procesar imágenes se debe tener el mayor cuidado, debido a que, ciertas funciones de OpenCV suelen cambiar el formato de la imagen de RGB a BGR, dando una variación en los valores procesados de las fotos ingresadas frente a las utilizadas en el modelo clasificador, y al realizar un escalado de imagen sin un control se puede deformar sus características que, al momento de obtener los valores de su matriz, estos se verán afectados en comparación con una foto bien escalada.

Para mejorar la predicción en un futuro se podría realizar un modelo personalizado y enfocado en canes para la obtención de incrustaciones, se debe de tener en cuenta que una de las desventajas es la limitación de recursos informáticos y la complejidad para construir un modelo más robusto con aprendizaje profundo. Además, para mejorar las técnicas utilizadas se debe comprobar el rendimiento al evaluar un dataset más extenso, caso contrario, se podría trabajar con otros clasificadores que sean apropiados para la clasificación de más de dos clases.

El software desarrollado puede servir como una guía para continuar con la investigación, otorgando mejoras en el código o con la implementación de nuevos métodos, con la finalidad de que el programa sea más eficiente al momento de hacer pruebas con datasets reales (otorgados por la sociedad), con el fin de comprobar tiempos de ejecución y eficiencia, para determinar si el programa puede ser utilizado en tiempo real o si entregará un resultado en cuestión de minutos.

REFERENCIAS

[1] ARCA, «Plan sistematizado para el manejo de animales domésticos de compañía del cantón Cuenca.,» Cuenca, 2018.

[2] J. Raisen, «American Kennel Club,» 14 Abril 2021. [En línea]. Available: https://www.akc.org/expert-advice/lifestyle/how-do-dog-microchips-work/. [Último acceso: 30 Julio 2021].

[3] N. Dávalos, «Primicias,» 23 Noviembre 2019. [En línea]. Available: https://www.primicias.ec/noticias/tecnologia/es-del-tamano-de-un-arroz-y-va-bajo-la-piel-asi-es-la-cedula-para-su-mascota/. [Último acceso: 5 Agosto 2021].

[4] V. Ribero, «Uso de códigos bidimensionales y posicionamiento para el reencuentro de mascotas con sus dueños.,» La Plata, 2016.

[5] J. Polimeno, «Facial recognition lost pet identifying system,» California, 2016.

[6] K. Lai, X. Tu y S. Yanushkevich, «Dog Identification using Soft Biometrics and,» Beijing, 2019.

[7] D. Batic y D. Culibrk, «Identifying Individual Dogs,» Novi Sad, 2020.

[8] P. Rooyakkers, «Method and system for alerting an owner of a lost animal,» Vancouver, 2014.

[9] H. Hosseini, «Animal muzzle pattern scanning device,» Los Angeles, 2015.

[10] D. Crouse, R. Jacobs, R. Zach, K. Scott, J. Anil, B. Andrea y T. Stacey, «LemurFaceID: a face recognition system to facilite individual identification of lemurs,» BMC Zoology, 2017.

[11] E. Bassaure, V. Cárdenas y C. A. Hernández, «Reconocimiento facial en perros, un modelo de vigilancia beneficioso,» DESARROLLO E INNOVACIÓN CLIDi, 2019.

[12] G. Mougeot, D. Li y S. Jia, «A Deep Learning Approach for Dog Face Verification and Recognition.,» PRICAI 2019: Tendencias en Inteligencia Artificial, 2019.

[13] E. Baxter, M. F.Hansena, M. L.Smith, L. N.Smith, M. G.Salter, M. Farish y B. Grieve, «Towards on-farm pig face recognition using convolutional neural networks,» Computers in Industry, vol. 98, pp. 145-152, 2018.

[14] L. He, «Insurtech giant ZhongAn plans to use facial recognition, blockchain to monitor chickens,» South China Morning Post, 2017. [En línea]. Available: https://www.scmp.com/business/companies/article/2123567/blockchain-and-facial-recognition-zhongan-techs-recipe-changing.

[15] Google, «Iniciar en Google Colab,» [En línea]. Available: https://colab.research.google.com/signup#.

[16] G. P. A. d. l. E. Enrique Alegre, Conceptos y Métodos en Visión por Computador, España: Comité Español de Automática CEA, 2016.

[17] A. W. F. E. Lavin J. Halawa, «Face Recognition Using Faster R-CNN with Inception-V2 Architecture for CCTV Camera,» IEEE, 2019.

[18] J. M. F. Raquel Flórez López, Las Redes Neuronales Artificiales, 2008.

[19] S. Shetty, «Packt,» 18 Septiembre 2018. [En línea]. Available: https://hub.packtpub.com/what-is-pytorch-and-how-does-it-work/. [Último acceso: 8 Abril 2021].

[20] S. Ray, «Analytics Vidhya,» 13 Septiembre 2017. [En línea]. Available: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/. [Último acceso: 21 Abril 2021].

[21] Hazelcast, Hazelcast Co, San Mateo, 2020.

[22] J. A. Rodrigo, «Ciencia de Datos, Estadística, Machine Learning y Programación,» Abril 2017. [En línea]. Available: https://www.cienciadedatos.net/documentos/34_maquinas_de_vector_soporte_support_vector_machines#DAGSVM. [Último acceso: 29 Julio 2021].

[23] L. Moreno, «La biometría en el mundo animal: Los sistemas de identificación biométrica animal son una tendencia prometedora.,» biometriaAplicada, 2019. [En línea]. Available: https://biometriaaplicada.com/sitio/la-biometria-en-el-mundo-animal-los-sistemas-de-identificacion-biometrica-animal-son-una-tendencia-prometedora/.

[24] J. L. M. LOPEZ, ANATOMIA CLINICA DEL PERRO Y GATO, España, 2009.

[25] Galván y Ajo, «Reconocimiento facial para localizar a perros perdidos: la nariz canina tiene la clave para su identificación,» srperro.com, 2018. [En línea]. Available: https://www.srperro.com/consejos/curiosidades/reconocimiento-facial-para-localizar-a-perros-perdidos-la-nariz-canina-tiene-la-clave-para-su-identificacion/. [Último acceso: 4 Septiembre 2020].

[26] La Razón, «Recuperar a tu mascota a través del reconocimiento facial,» 25 Noviembre 2019. [En línea]. Available: https://www.larazon.es/familia/20191125/u2267f4c5veu3mqka735mguwaa.html. [Último acceso: 8 Septiembre 2020].

[27] P. C. G. Stanley H. Done, Atlas en color de anatomía veterinaria, 2010.

[28] D. Tzutalin, «LablImg,» 2 Agosto 2020. [En línea]. Available: https://github.com/pranjalAI/labelImg. [Último acceso: 24 Agosto 2021].

[29] J. Barrios, «Health Big Data,» 21 Junio 2020. [En línea]. Available: https://www.juanbarrios.com/google-machine-learning/. [Último acceso: 1 Septiembre 2021].

[30] T.-Y. Lin, P. Goyal, R. Girshick, K. He y P. Dollár, «Focal Loss for Dense Object Detection,» 2018.

[31] S. Ren, K. He, R. Girshick y J. Sun, «Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,» IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, nº 6, pp. 1137-1149, 2017.

[32] E. Amador, V. González y C. Martínez, «Reconocimiento facial en perro, un modelo de vigilancia beneficioso,» Ciudad de México, 2019.

[33] R. Thaware, «EEWeb,» 28 Mayo 2018. [En línea]. Available: https://www.eeweb.com/real-time-face-detection-and-recognition-with-svm-and-hog-features/#:~:text=A%20HOG%20is%20a%20feature%20descriptor%20generally%20used%20for%20object%20detection.&text=In%20the%20current%20example%2C%20all,per%20pixel%20of%20the%20im. [Último acceso: 20 Abril 2021].

[34] A. Singh, «Analytics Vidhya,» 4 Septiembre 2019. [En línea]. Available: https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/. [Último acceso: 20 Abril 2021].