ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Sistema inteligente para el control, cuidado y comportamiento de animales en rehabilitación y cautiverio

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Ingeniero en Telemática

Presentado por:

Guillermo José Sánchez Guzmán

Joshua Israel Taranto Usey

GUAYAQUIL - ECUADOR

Año: 2022

DEDICATORIA

GUILLERMO JOSÉ SÁNCHEZ GUZMÁN

El presente proyecto lo dedico a mis padres y a mi hermana, quienes estuvieron apoyándome en todo momento, incluso cuando yo dejaba de creer en mi. También lo dedico a la memoria de mi abuelito Simón Guzmán, de mi tío Mauricio Sánchez y de Sabrina porque a pesar de no tenerlos físicamente conmigo, siempre sentí su apoyo.

JOSHUA ISRAEL TARANTO USEY

A aquellos que nunca perdieron la fe en mí, incluso en los momentos en los que yo mismo dudaba, les dedico este trabajo con profundo agradecimiento. Gracias por ser mi pilar de fortaleza y por empujarme a seguir adelante.

AGRADECIMIENTOS

GUILLERMO JOSÉ SÁNCHEZ GUZMÁN

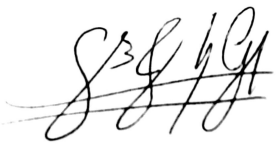
Agradezco a mis padres por su apoyo incondicional, a los profesores que tuve desde mi etapa de escuela por la formación que me han brindado. A toda mi familia por siempre haber creido en mi y expresarme su orgullo. A mi grupo de amigos, los estimados, y a Melanie, por haberme alegrado en mis momentos difíciles. A Jessica y su familia por haberme ayudado durante toda mi etapa de universidad. Al Doctor Washington Velásquez a quien considero como mi mentor por haberme brindado el conocimiento que más valoro de mi carrera.

JOSHUA ISRAEL TARANTO USEY

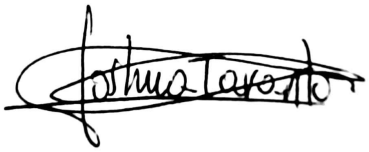
A todas aquellas personas que siempre estuvieron ahí para brindar su apoyo y aliento, les doy las gracias más sinceras. Su presencia en los momentos buenos y malos nunca será olvidada y será siempre apreciada. Gracias por su amor y apoyo incondicional.

DECLARACIÓN EXPRESA

”Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; Guillermo José Sánchez Guzmán y Joshua Israel Taranto Usey y damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual”



Guillermo José Sánchez Guzmán



Joshua Israel Taranto Usey

EVALUADORES

Firmado electrónicamente por:
WASHINGTON ADRIAN
VELASQUEZ VARGAS

WASHINGTON VELÁSQUEZ, PhD.

PROFESOR DE LA MATERIA Y TUTOR

RESUMEN

La situación de los animales callejeros en Ecuador preocupa tanto a aquellos que luchan por sus derechos como a la población en general. El llevarlos a un refugio no basta, pues, debido a la alta población de mascotas que son abandonadas, no es posible acogerlos a todos y brindarles los cuidados necesarios ya sea por falta de personal o de recursos.

El objetivo principal de este trabajo es detectar los estados de ánimo de los animales captados en cámara. Para esto, se usan técnicas de análisis de imagen para identificar los patrones de comportamiento de los animales. A través de los resultados obtenidos por medio del desarrollo, se pudo determinar que el sistema podría resultar efectivo si se lleva a ambientes no controlados como con el que se trabajó. Mediante las encuestas se logró determinar que la propuesta puede ser muy útil para los refugios de animales, veterinarios y dueños de mascotas que no poseen el tiempo o los medios para estar al pendiente las 24 horas de sus mascotas y necesitan, de alguna manera, agilizar dicha tarea.

Este documento se encuentra dividido en 5 capítulos. En el capítulo 1 se explica el problema que se trata de resolver, junto con los objetivos, el alcance, las limitaciones, y el marco teórico. El capítulo 2 describe la metodología aplicada, la arquitectura del proyecto, y los materiales usados para la construcción. En el capítulo 3 se presenta el diseño de la solución y la implementación, se describe todo el proceso que se siguió para que se pudiera concretar el proyecto y cómo las arquitecturas usadas interactuaron entre sí para cumplir con los objetivos planteados. En el capítulo 4 se analizan los resultados generados para determinar la efectividad del proyecto. Por último, en el capítulo 5 se describen las conclusiones, recomendaciones y las líneas futuras sobre el trabajo desarrollado.

Palabras Clave: Animales, Modelos, Refugios, Ánimo, Veterinarios

ABSTRACT

The situation of stray animals in Ecuador concerns those who fight for their rights and the general population. Taking them to a shelter is not enough because, due to the high population of abandoned pets, it is impossible to accommodate them all and provide all the necessary care, either due to a lack of personnel or resources.

The main objective of this work is to detect the moods of the animals captured on camera. For this purpose, image analysis techniques are used to identify the behavioral patterns of the animals. The results obtained employing the development made it possible to determine that the system could be effective if taken to uncontrolled environments such as the one we worked with. The surveys determined that the proposal could be beneficial for animal shelters, veterinarians and pet owners who do not have the time or the means to be aware of their pets 24 hours a day and need, in some way, to expedite this task.

This document is divided into 5 chapters. Chapter 1 explains the problem to be solved, along with the objectives, scope, limitations, and theoretical framework. Chapter 2 describes the methodology, project architecture, and materials used in the development process. Chapter 3 presents the design of the solution and the implementation, it is described the whole process that was followed so that the project could be realized and how the architectures used interacted with each other to meet the objectives set. Chapter 4 analyzes the obtained results to determine the effectiveness of the project. Finally, Chapter 5 describes the developed project's conclusions, recommendations, and future lines of action.

Keywords: Animals, Models, Shelters, Mood, Veterinary

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE CODIGOS DE PROGRAMA

CAPÍTULO 1

1. INTRODUCCIÓN

El abandono de animales en Ecuador es una problemática que cada vez tiene más acentuación e inclusive se ha agravado notoriamente con la llegada de la pandemia del SARS-CoV-2 (COVID 19)[1] y la crisis económica mundial. Un estudio realizado por la Fundación Affinity titulado “El nunca lo haría” [2] enfocado a la pérdida, el abandono y la adopción de animales en España, 2020, establece que durante ese año los centros de acogida de animales de compañía de ese país recibieron aproximadamente 162.000 perros y 124.000 gatos. En Ecuador, a pesar de que existen leyes que se encargan de regular la tenencia de mascotas como la Ley Orgánica de Bienestar Animal (LOBA) y el Código Orgánico del Ambiente, el aumento del abandono de perros dificulta la sostenibilidad urbana, el equilibrio de la vida silvestre en zonas urbanas y rurales y la seguridad de la ciudadanía en general. La Organización Mundial de Sanidad Animal, (2016), indica que en muchos países del mundo, los perros callejeros constituyen una carga para la sociedad. Estos animales representan un papel importante en la transmisión de enfermedades zoonóticas como la rabia; también en la contaminación fecal o acústica; en los riesgos para el bienestar de las personas debido a las mordeduras o atropellos, y en los riesgos para otros perros o especies de animales domésticos, salvajes o de producción.

Debido al problema existente de los ataques por parte de animales callejeros a personas, así como el poco entendimiento que hay por parte de las personas hacia los animales, este proyecto tiene como enfoque implementar un sistema automatizado de monitoreo mediante hardware y software libre para el control, cuidado y comportamiento de animales en rehabilitación y cautiverio, evitando que se susciten problemas de ataques por parte de animales que deriven en consecuencias fatales para estos.

[1] https://www.lahora.com.ec/pais/el-abandono-de-perros-en-quito-se-incremento-en-90/

En este capítulo se analiza a profundidad el tema presentado, exponiendo cuáles son los estudios existentes relacionados, explicaciones que permitan comprender de mejor manera a los animales que se estudian. Se describe el problema a tratar, la justificación que permite entender el por qué es importante un sistema como este, y los objetivos y el alcance y las limitaciones, finalmente, el marco teórico presenta toda la investigación existente en torno a los temas que se tratan en este trabajo.

1.1 Descripción del problema

Según un estudio realizado por la Universidad San Francisco de Quito en 2018 [2], se estima que hay 600 mil animales de compañía en situación de calle. La mayoría de estos animales tienen la suerte de ser acogidos en refugios que les pueden proveer de alimento, hogar y una buena atención. A pesar de esto, los ataques por parte de mascotas o animales callejeros representan una problemática en varios países principalmente Latinoamericanos. Estos ataques suelen darse ya que las víctimas no pueden leer el lenguaje corporal del animal cuando se siente amenazado o incomodado. Debido a estos ataques muchos animales suelen ser sacrificados pues se los califica como un peligro para la sociedad. Los animales abandonados suelen pasar largas temporadas en refugios cuando son rescatados, eso sin contar el tiempo que podrían pasar en rehabilitación cuando tienen limitaciones físicas o traumas debido a algún accidente.

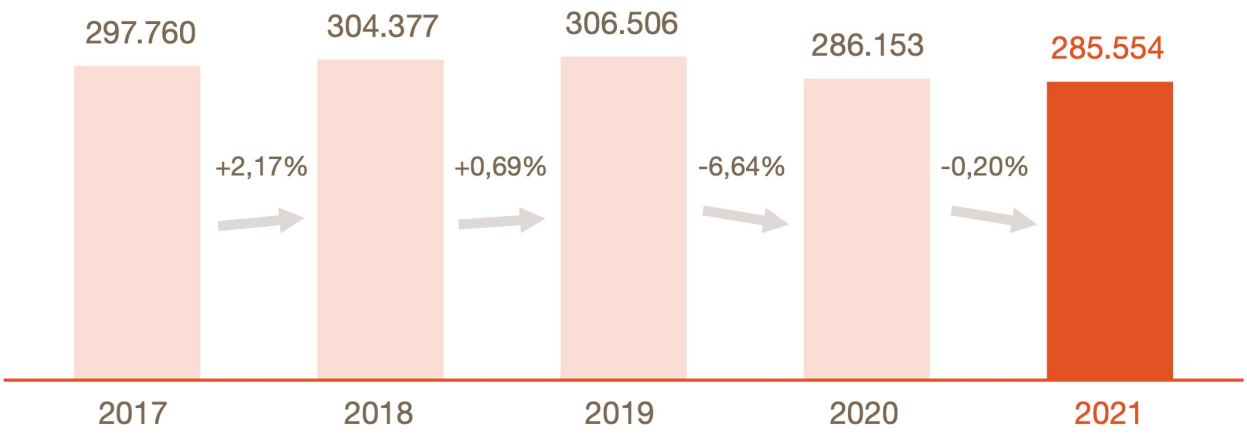


Figura 1.1: Cantidad de animales que llegan a refugios cada año [3]

[2] https://noticias.usfq.edu.ec/2018/09/resultados-del-1er-censo-ciudadano-de.html
[3] https://www.fundacion-affinity.org/observatorio/infografia-el-nunca-lo-haria-abandono-adopcion-perros-gatos-espana-2022

En Ecuador no hay estudios previos acerca del abandono de animales en el país en general, sin embargo, de acuerdo a un estudio realizado por la fundación affinity en España [4], durante el año 2021 llegaron a refugios una cantidad de 285,554 animales, en contraste con el año 2017 que fueron 297,760, como se observa en la figura 1.1. El descenso de la cifra del año 2021 es moderado con respecto al año 2020, sin embargo es aún una cifra muy alta, teniendo 285,554 animales, entre los cuales se encuentran 167,656 perros y 117,898 gatos.

1.2 Justificación del problema

Actualmente los derechos de los animales es un tema que ha obtenido aceptación por parte de la ciudadanía, existen muchos países que ya reconocen el respeto a los derechos de los animales llegando a penalizar la violación de los mismos, e incluso haciéndolos parte de su constitución, sin embargo, aún hay muchos aspectos por pulir al respecto.

Los animales en cautiverio pueden pasar mucho tiempo sin encontrar a alguien que los adopte, por lo que el lugar en el que son mantenidos en cautiverio se convierte en su hogar. Debido a que son animales rescatados, muchos presentan comportamientos poco comunes en una mascota típica y que podrían resultar peligrosos para alguien no experto. A causa de esto es indispensable una herramienta que permita analizar el comportamiento de un animal, porque resulta muy útil detectar el estado en el que se encuentra sólo por medio de un análisis físico del mismo usando una imagen.

El sistema que se desarrolla consta de una cámara que sirve para capturar las imágenes que se analizarán, un servidor que aloja la aplicación web, y una computadora que realiza el entrenamiento del modelo y el análisis de la imagen.

[4] https://www.lahora.com.ec/pais/el-abandono-de-perros-en-quito-se-incremento-en-90/

1.3 Objetivos

1.3.1 Objetivo General

- Implementar un sistema automatizado de monitoreo mediante hardware y software libre para el control, cuidado y comportamiento de animales en rehabilitación y cautiverio.

1.3.2 Objetivos Específicos

- Adaptar un algoritmo de aprendizaje de máquina mediante las librerías *Tensorflow* y *Keras* para el reconocimiento de imágenes de animales.
- Predecir el estado de ánimo de un animal utilizando imágenes capturadas en tiempo real para la ejecución/activación de eventos físicos de cuidado.
- Implementar una aplicación de monitoreo utilizando un marco de referencia web para los cuidados de animales en cautiverio o rehabilitación.

1.4 Alcance Y Limitaciones

El propósito de este escrito es presentar una propuesta para que los refugios de animales, clínicas veterinarias y dueños de mascotas puedan garantizar la seguridad y el bienestar de los animales. Se propone un sistema que monitorea en tiempo real a un animal con el fin de predecir su estado de ánimo y mostrarlo en una página web junto con una alarma que se enviará a un asistente inteligente. Las especies escogidas para realizar el estudio son perros y gatos, por lo que la recolección de datos se enfoca solo en dichos animales. Las razas en las que se centra el estudio son el *Golden Retriever* y el *Gato Azul*. Se tienen dos algoritmos entrenados uno para cada especie por facilidad del desarrollo del proyecto. La principal limitación del sistema es que el proceso se realiza en un entorno completamente controlado y no con expectativas de resultados críticos, es decir, la mascota monitoreada no padece de enfermedad alguna.

1.5 Estado del Arte

Los animales callejeros en el mundo han causado el surgimiento de muchos movimientos que defienden sus derechos. En 1978 se publicó el primer texto sobre la Declaración Universal de los derechos del animal, el cual fue promovido por la Liga Internacional de los Derechos de los Animales. Tras varios cambios se envió el texto en repetidas ocasiones a la *UNESCO* para que sea aprobado. Sin embargo, aún no se ha materializado [3]. La iniciativa más cercana y relevante del desarrollo del Derecho Animal vio frutos el 18 de diciembre del año 2014 en Argentina [4]. En el marco de lo mencionado, cabe destacar que para la mayoría de las personas resulta complicado entender a un animal, como se menciona en [5], el problema de las personas es que son demasiado cerebrales. A pesar de que existen leyes que protegen los derechos de los animales, resulta difícil hacer reformas debido a que la mayoría de aquellos que rigen los organismos oficiales que regulan las leyes nunca han estado en el interior de una empresa de productos cárnicos, el desconocimiento acerca del trato que reciben los animales en muchos de estos lugares no permite crear empatía hacia estas situaciones. Aunque existe la barrera entre el humano y el animal para poder comprender lo que se quiere expresar, se han propuesto herramientas que se encargan de hacer esta tarea un poco más sencilla. En [6] se desarrolla un sistema que permite conocer el nivel de dolor que sienten las ratas al hacer un análisis de sus expresiones faciales. Luis Corujo et al. [7], presenta un sistema de reconocimiento de emociones en caballos usando redes neuronales convolucionales. Existen trabajos que se encargan de realizar tareas similares a este proyecto, sin embargo, no se tiene algo tan general como lo que se presenta, para tener un sistema más general hace falta primero analizar a los sujetos de estudio y cómo estos interactúan con el mundo que les rodea.

De acuerdo a [8] durante el 2008 y 2009 habían alrededor de 312.500 perros en el Distrito Metropolitano de Quito, de los cuales 112.500 se encontraban en situación de calle, además, 12.500 tenían dueño que los dejaban pasear por espacios públicos. Según datos de la fundación PAE, se reciben 350 perros cada mes, de los cuales sólo 92 son adoptados. En este contexto, José Paredes [9] estudió a 668 perros deambulado en Quito y todas sus parroquias de los cuales la mayoría son machos, lo cual puede ser a causa de que las hembras son más vulnerables en estos ambientes, pues la reproducción, es

un factor predisponente para la disminución de estas. El mayor número de los perros observados eran mestizos. Se menciona además que la agresividad en los perros se puede clasificar en dos grupos:

- **Con causa orgánica:** agresividad causada por dolor.
- **Sin causa orgánica:** agresividad por competencia, la cual se presenta mayormente en los machos adultos con el fin de ser dominantes. Los perros agresivos con impulsividad, tienen la particularidad de no dar señales previas a un ataque, ya sea gruñidos, o fruncir los belfos. La agresividad por miedo se presenta con la misma frecuencia tanto en hembras como en machos, ante una situación o ante personas desconocidas muestran cierto temor, la agresividad por miedo hacia personas se debe a la falta de una socialización.

1.5.1 Sistemas de reconocimiento de emociones en animales

Hasta la fecha, la investigación enfocada en usar técnicas computacionales para el reconocimiento de las emociones en animales es escasa. Los resultados de [10] demuestran que los conjuntos de características acústicas diseñados para capturar las emociones humanas se pueden utilizar para clasificar la intensidad, la emoción percibida o el contexto del ladrido de un perro haciendo uso de *EGEMAPS* [11]. Así mismo, Valentina Franzoni et. al. [12] demuestra que es posible que un sistema de aprendizaje de máquina, construido con una red neuronal convolucional pre-entrenada, logre reconocer las emociones de los animales de la misma manera en que los humanos las perciben. Los resultados preliminares muestran que el enfoque es factible y que el sistema es capaz de reconocer en imágenes de perros lo que los humanos identifican comúnmente como emociones de perro. Se obtuvieron resultados muy satisfactorios del clasificador, teniendo un promedio de 86.96 por ciento de decisiones fuertemente correctas, y solo 5.43 por ciento débilmente correctas de acuerdo a las métricas usadas en el estudio. Por otro lado, Richard O. Sinnott et. al. [13], explora el uso de Deep Learning para detectar las emociones en mascotas, haciendo uso de imágenes de 52 razas de perros y 23 razas de gatos. La librería *YOLOv5* sirvió para desarrollar el modelo, el resultado de dicho proyecto fue una aplicación móvil y según la retroalimentación de los usuarios, se obtuvo niveles de precisión considerables en la clasificación de las emociones de perros y de gatos como

se observa en las figuras 1.2 y 1.3 en las cuales se generaron estadísticas acerca de la precisión de la aplicación al identificar los distintos estados en las mascotas, como se puede observar, se obtuvieron mejores resultados para cuando el animal se encuentra feliz, mientras que en los casos de ansiedad y tristeza la diferencia de proporciones es baja.

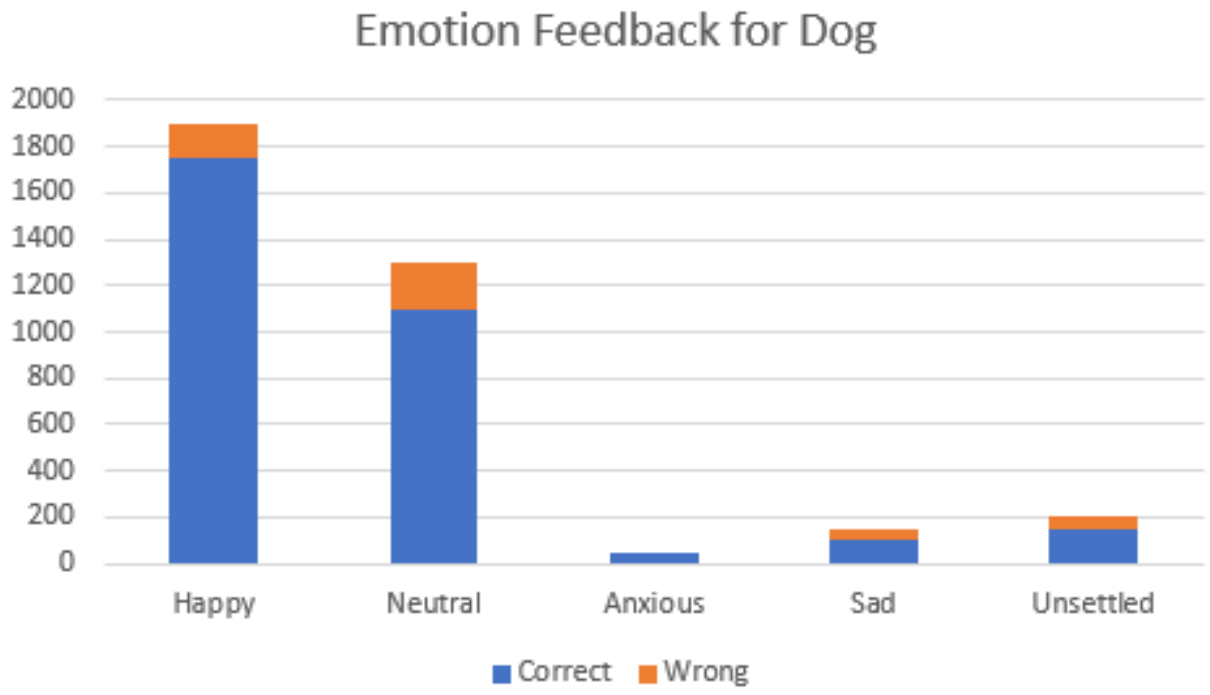


Figura 1.2: Precisión en la clasificación de emociones en perros

1.5.2 Lenguaje corporal

1.5.2.1 Lenguaje Corporal de los Perros

S. Coren y S. Hodgson [1] explican cómo entender a los perros, se hace especial énfasis en el lenguaje corporal de los perros y cómo estos transmiten muchas de sus emociones solo por medio de sus movimientos. En las figuras 1.4 y 1.5 se explica cómo identificar comportamientos de agresividad y sumisión en los perros.

Se debe hacer especial énfasis en la cola. En el primer caso, la cola levantada demuestra una actitud dominante, mientras que en el segundo caso, la cola escondida entre las patas denota una sensación de temor por parte del perro. Se puede observar en 1.6 la comparación que se hace de la altura de la cola con respecto a la actitud del

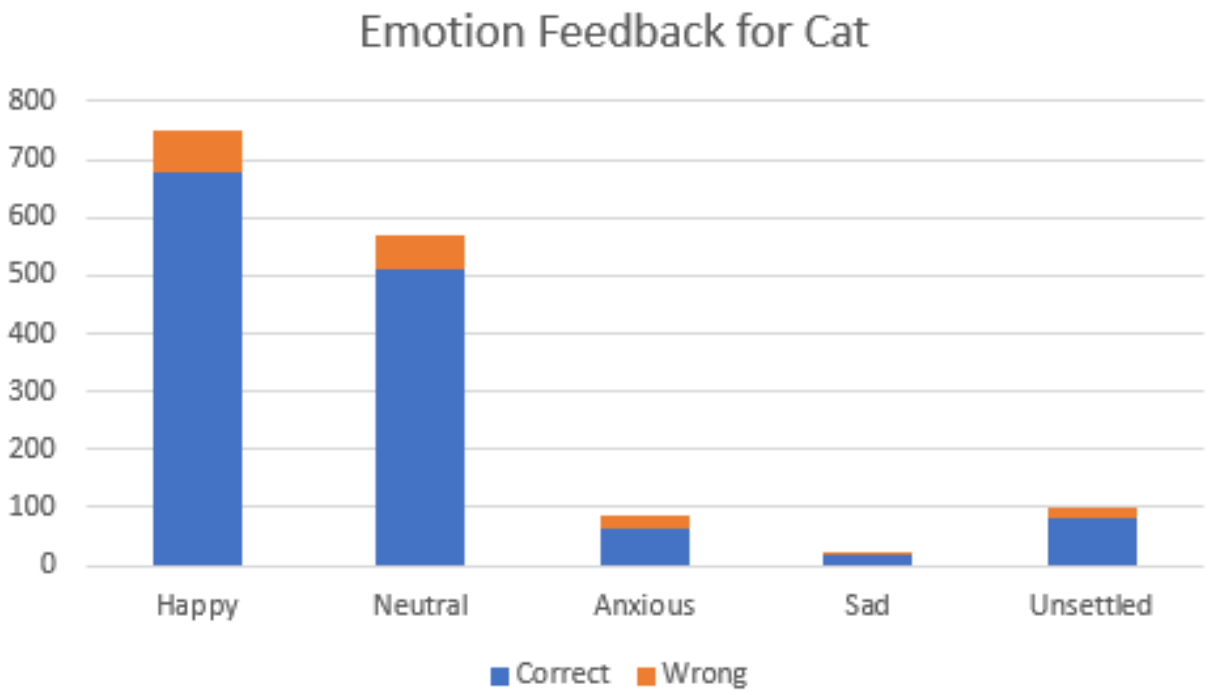


Figura 1.3: Precisión en la clasificación de emociones en gatos

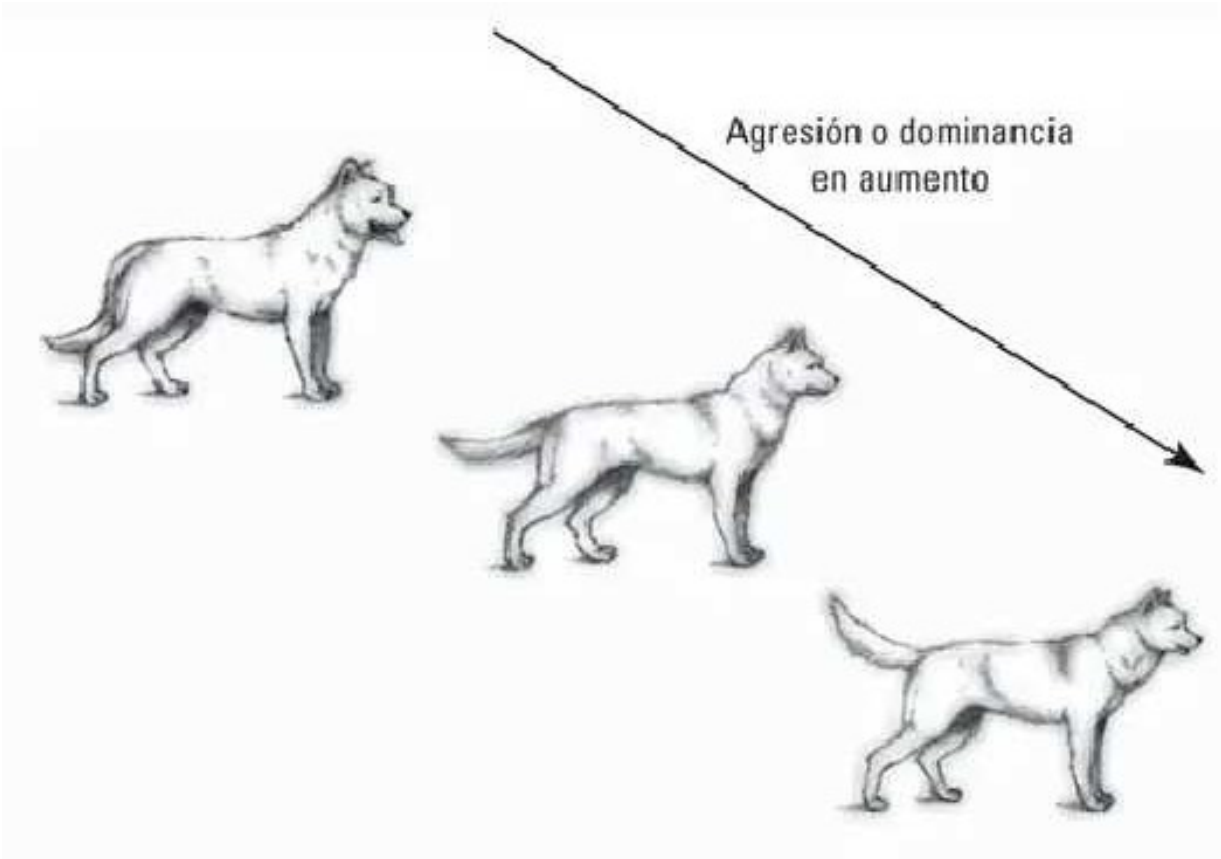


Figura 1.4: Posturas que denotan agresividad y dominancia [1]

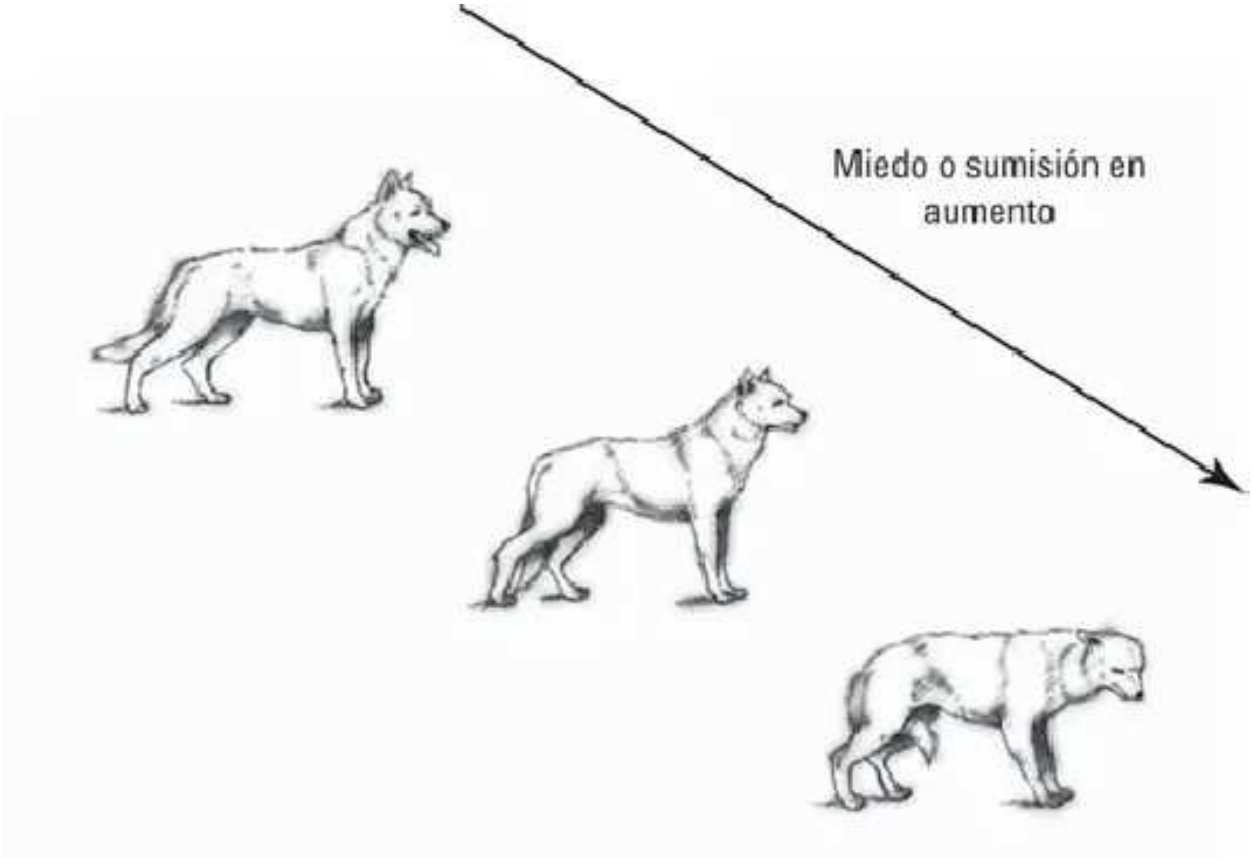


Figura 1.5: Posturas que denotan temor y sumisión [1]

perro.

1.5.2.2 Lenguaje corporal de gatos

El lenguaje corporal, los sonidos que emiten y sus posturas son algunos de los comportamientos que los felinos usan para comunicar sus emociones [14]. La raza y la colonia influyen mucho en la comunicación de estos animales. En un ambiente con muchos gatos sus comportamientos pueden variar dependiendo del área, los recursos alimenticios y de quien esté presente. Los gatos necesitan comunicarse entre sí para vincularse y relacionarse; necesitan colaborar, jugar y compartir recursos. Cuando se comunican con las personas, lo hacen para obtener lo que necesitan o desean, como comida, afecto o juegos [15].

Los felinos transmiten a través de gestos y ciertas acciones indicando varias emociones tales como peligro, amenaza, tristeza o cuando tienen hambre. Por ejemplo, cuando llega un nuevo gato a su territorio, se muestran amenazados alborotando su pelaje, arqueando su lomo e incluso haciendo gruñidos [16] para imponer su área. Los

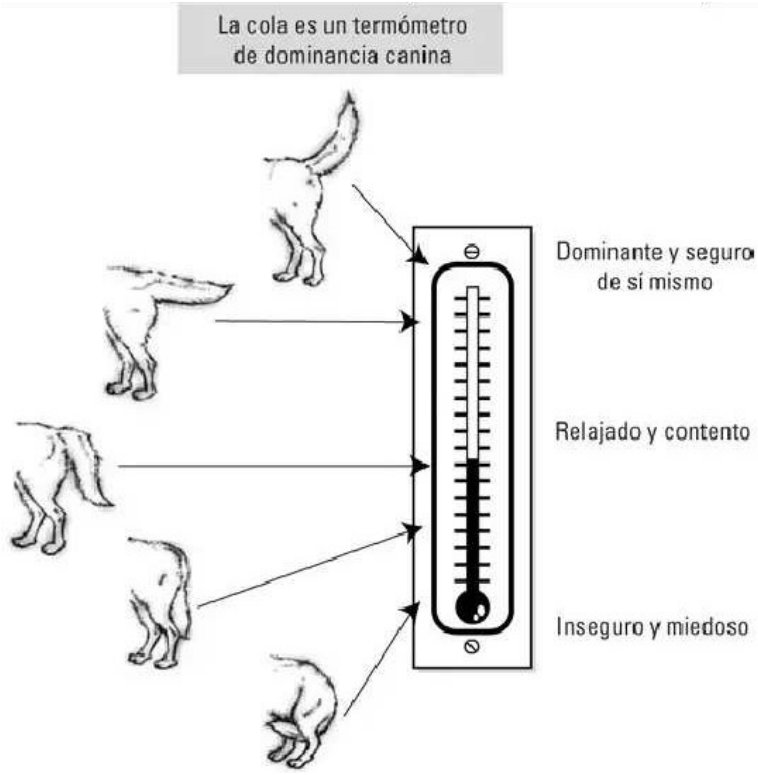


Figura 1.6: Posiciones de la cola [1]

gatos son animales que han sido domesticados a lo largo de los años [17] provocando así su desarrollo en la comunicación con los humanos [18] a través de sus gestos faciales [19].

Postura de gatos - Existen varias posturas que indican las emociones de un gato y tales emociones pueden variar dependiendo de quienes estén a su alrededor. Lo ideal es analizar la emoción de un gato cuando se encuentre solo. Las posturas más comunes entre los felinos son ante alguna amenaza [20]. En esta postura el gato se pone tenso encorvando su torso y alborotando su pelaje junto a un gruñido para imponer territorio. Otra postura muy popular entre los gatos, es bostezando; esto indica que el gato se está relajando. Es común que mientras se están bostezando incluyan un estiramiento. Cuando un gato acerca su torso ante alguna superficie, significa que esta temeroso ante algún animal o situación poco familiar. La postura de las orejas aplanadas indica que el felino se encuentra en posición de defensa o a su vez, de ataque [21].

Comunicación visual en gatos - Los gatos suelen comunicar mucho con los ojos [20]. Cuando un felino parpadea lento mientras mira fijamente, es sinónimo de confianza. Según Gary Weitzman, veterinario licenciado y autor de animales, este tipo de lenguaje corporal felino es similar a un ”beso de gatito”. Él explica además en su libro, ”Cómo

hablar gato: una guía para decodificar el lenguaje de los gatos" [22], que el parpadeo lento podría ser una respuesta fisiológica a la reducción de los niveles hormonales de estrés por estar en un estado de calma.

Posición de la cola de un gato - Otra manera en que los gatos se comunican es a través de la posición de su cola. Cuando su cola esta casi a 90º se relaciona con que el gato se siente feliz o en confianza, mientras que si la cola esta aproximadamente a 45º indica que el gato no está muy contento y si esta la cola abajo indica infelicidad [23]. Los gatos reflejan su relajamiento de muchas maneras, una de estas es agitando su cola de derecha a izquierda. Aunque si mueven su cola en forma de serpiente, indican que están indecisos o quieren acechar algo [24].

1.5.3 Trabajos Relacionados

De acuerdo a la investigación realizada por Suresh Neethirajan [25], se pueden obtener mejores resultados en el reconocimiento de las emociones si se realiza un análisis de las expresiones faciales del sujeto, para lo cual se deben tomar las fotografías necesarias en un cuadro que permita obtener por completo lo que expresa el rostro del animal, e.g., en [26] se indica que es necesario un ambiente con buena iluminación para poder realizar un reconocimiento facial que capture todos los detalles, esto podría sustituirse por una cámara con buena resolución, sin embargo, se debe capturar la mayor cantidad de minucias de un rostro para poder definir con alta precisión el estado que expresa el rostro. Por otro lado, en [27] se somete a un modelo de reconocimiento facial a pruebas de estrés y se lo testea bajo condiciones diferentes con el fin de analizar su efectividad, se descubrió que se tiene mayor efectividad cuando se hace uso de luz natural, cuando se tienen accesorios como aretes la efectividad disminuye dependiendo del área del rostro que cubra el accesorio, el ángulo de inclinación también tiene mucha influencia al momento de generar resultados pues al aumentar la inclinación de la cámara, se reduce la efectividad. Finalmente, cuando se intenta realizar el reconocimiento con el perfil del rostro, el algoritmo falla. Como se sugiere en [28], el Algoritmo de Viola Jones es muy eficaz para realizar reconocimiento pues realiza la clasificación mediante características extraídas en una escala de grises, por lo que este algoritmo podría representar un punto de partida o al menos, de referencia, para la elaboración de este proyecto

La investigación en los temas tratados no es abundante, sin embargo, resulta

suficiente como punto de partida para este proyecto y lo que se quiere lograr. La diferencia de este proyecto es que se implementará todo un sistema completo el cual comprende el monitoreo en tiempo real de los sujetos, el análisis de sus estados de ánimo y generar un resultado a partir de dicho análisis, el resultado se podrá visualizar por medio de la aplicación web a la cual el veterinario o encargado de refugio tendrá acceso.

CAPÍTULO 2

2. Metodología

En este capítulo se relata la metodología, los materiales y mecanismos para el desarrollo del proyecto. El trabajo se lleva a cabo siguiendo una estructura segmentada de manera estratégica con el fin de cumplir los objetivos planteados [1]. Para este proyecto se utiliza el método de investigación aplicada pues se pretende resolver un problema tomando como base la búsqueda y consolidación del conocimiento para que se pueda aplicar. El sistema propuesto hace uso de las imágenes recopiladas de animales para alimentarse y ser entrenado. Se empieza explicando la arquitectura del sistema y cada una de sus secciones. Posteriormente se analizan los criterios de inclusión y exclusión que se usarán. Finalmente se habla sobre las métricas para determinar la efectividad del producto entre las que se encuentran las métricas de evaluación de la plataforma y las de satisfacción del usuario.

2.1 ¿Es posible realizar el cuidado de animales con el enfoque propuesto?

Resulta importante una herramienta que permita monitorear en tiempo real a los animales en un refugio o en una veterinaria con el fin de asegurar su bienestar de una manera eficiente, es por esto que mediante una página web y haciendo uso de Django se podrá realizar dicha función; Django se usará para la construcción de la página debido a que ofrece un despliegue sencillo y un fácil enlace con la base de datos Postgresql la cual almacenará toda la información generada por las lecturas y los usuarios. En el contexto del reconocimiento de emociones, el uso de Machine Learning ofrece un medio que

[1] https://bibliotecas.duoc.cl/investigacion-aplicada/definicion-proposito-investigacion-aplicada

permite reconocer los patrones físicos que se expresan cuando se genera algún tipo de emoción. Por esto, implementar una página web que se enlace con la función de Machine Learning junto con el monitoreo en tiempo real de los animales con el fin de detectar la emoción que están expresando será importante en el mundo del cuidado animal. La arquitectura que se propone permite la comunicación entre las partes logrando que la información que alimenta al modelo se transmita en tiempos muy cortos, es por esto que resulta muy efectivo para brindar cuidados a los animales, pues al recibir alarmas al instante de que se realiza la lectura del estado del sujeto, se podrá tomar las acciones correspondientes en caso de que se encuentre hambriento o estresado.

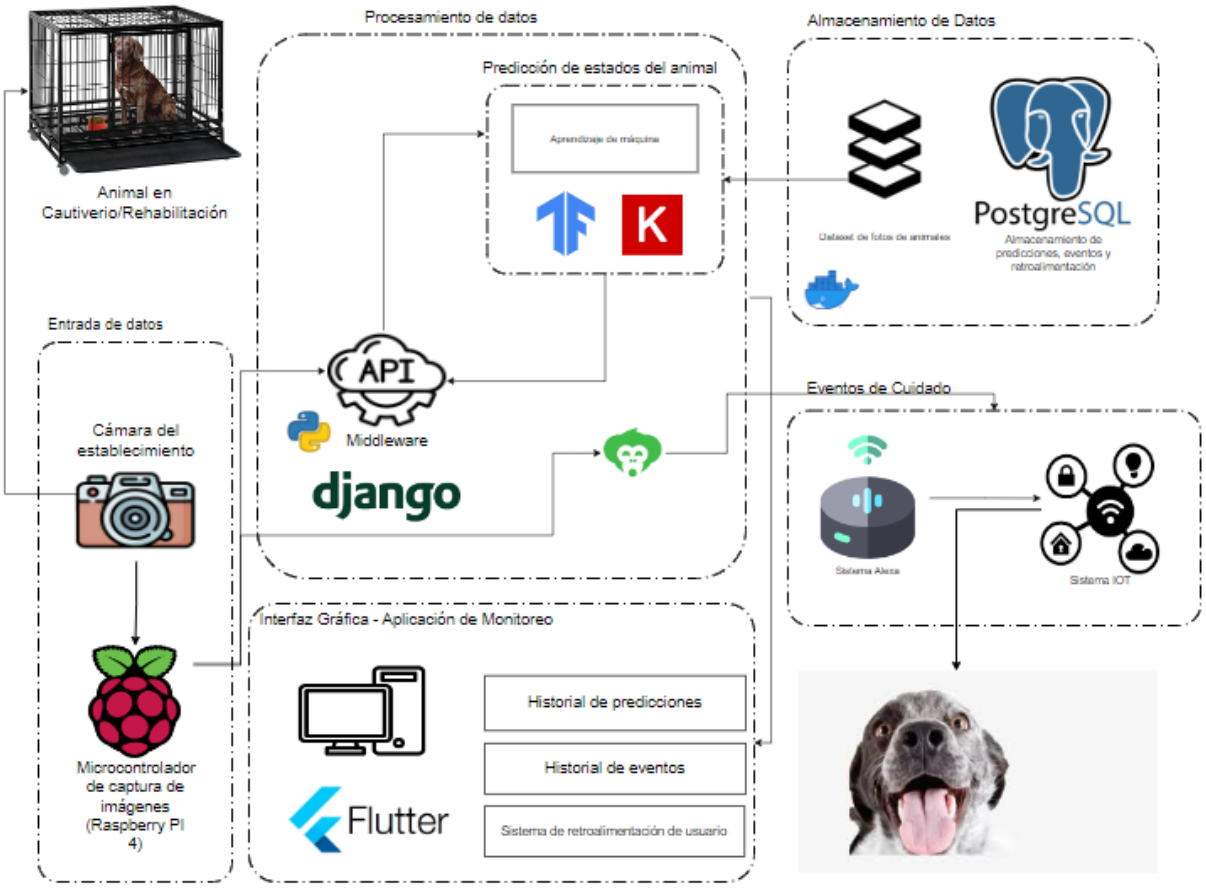


Figura 2.1: Arquitectura del Sistema

2.2 Arquitectura del proyecto

Para comprender la comunicación entre las distintas tecnologías que se usan para implementar el proyecto se construyó un diagrama 2.1 que sirve como referencia visual

de la arquitectura, la cual se segmenta por bloques de acuerdo a la función de cada uno.

La arquitectura se ha dividido en: Entrada de datos, procesamiento de datos, interfaz de usuario, almacenamiento de datos, eventos de cuidado.

2.2.1 Entrada de datos

Una cámara que monitorea al animal durante las 24 horas es colocada. A la cámara se le conecta una *Raspberry Pi 4* [2] que captura imágenes en formato *.jpg* cada cierto intervalo de tiempo. Este microcontrolador envia la imagen capturada hacia el servidor por medio de una API creada en *Django* [3].

2.2.2 Procesamiento de datos

En este proyecto, el procesamiento de datos se refiere al trato que se le da a la información ingresada al sistema, es decir, la imagen que se captura con la cámara para poder ser analizada por el modelo de aprendizaje automático y almacenada en la base de datos. El *middleware* se encarga de recibir la petición *HTTP* que contiene la imagen serializada en bytes realizada por la *Raspberry Pi 4*, luego se procede a guardar la imagen en el servidor que aloja la aplicación. Posteriormente se envía la fotografía hacia el modelo de aprendizaje de máquina, el cual se encarga de generar una respuesta acerca del estado del animal. La respuesta se almacena en la base de datos, conteniendo el estado detectado, la ruta de la imagen, y el animal al cual se le realizó el análisis.

2.2.3 Interfaz de usuario

El marco de trabajo *Flutter* [4] elaborado por Google es el utilizado para la implementación de la interfaz gráfica. Para cargar la información se consulta a la API de *Django*. Se le da acceso al usuario a una pantalla que muestra un resumen general de las predicciones, otra sección en la cual se puede observar la transmisión de la cámara utilizada para el monitoreo, se puede acceder al detalle de cada una de las predicciones y además una pantalla en la cual se puede dar valoraciones sobre las lecturas realizadas con el fin de

[2] https://www.raspberrypi.com/products/raspberry-pi-4-model-b/
[3] https://www.djangoproject.com
[4] https://flutter.dev

mejorar las predicciones al realizar un etiquetado más preciso de las imágenes tomando como guía la retroalimentación brindada por el usuario.

2.2.4 Almacenamiento de datos

2.2.4.1 Banco de imágenes

El banco de imágenes se constituye de fotografías de perros y gatos en las cuales se evidencian las emociones que se trata de predecir. Para la recopilación de datos se escogió a las siguientes razas: gato azul, labrador.

Los estados que se predicen son: hambre, felicidad, estrés.

Algunas de las imágenes se obtuvieron desde el sitio *Stanford Dogs Dataset*[5] en donde se agrupan por razas entre las cuales se encuentra la de labrador. Otras se obtuvieron tomando fotografías de mascotas cedidas de manera voluntaria por sus dueños con el fin de cumplir con este estudio. Inicialmente se trabaja con un dataset de alrededor de 20 imágenes por estado a detectar, es decir, 80 por animal, con lo cual se tienen 160 en total. Al finalizar se podrán encontrar las imágenes en una carpeta en Google Drive dentro de un archivo .zip.

2.2.4.2 Base de datos

Postgresql [6] es la base de datos elegida, pues funciona bien al conectarse con *Django*, lo cual facilita el enlace para la consulta e inserción de información. En la base de datos se almacenan:

- Los eventos ejecutados.
- El puntaje de las predicciones dado por los usuarios.
- Información relacionada a las predicciones como la fecha, la ruta de la imagen usada, y el resultado de la predicción.

[5] http://vision.stanford.edu/aditya86/ImageNetDogs/
[6] https://www.postgresql.org

2.2.5 Eventos de cuidado

Los eventos de cuidado permiten que se puedan dar los cuidados necesarios a los animales por medio de acciones que se realizan haciendo uso del hardware implementado y el sistema *Alexa*[7] como enlace para enviar las órdenes, estos eventos incluyen:

- Sintonizar música en un parlante comandado por *Alexa* para relajar al animal.
- Dar una alerta usando el sistema *Alexa* cuando el animal se siente hambriento para que el encargado llene su plato.
- Encender un foco que indica cuando el animal está feliz.

La integración de *Alexa* permite que la implementación del sistema se facilite al momento de enlazar el software con el hardware y ejecutar las acciones descritas anteriormente. Para generar los mensajes de voz dictados por Alexa se hace uso de *Voice Monkey* el cual es llamado por la raspberry y al activar un trigger Alexa da un mensaje enviado desde la plataforma.

2.3 Criterios de inclusión y exclusión

Los criterios de inclusión y exclusión sirven para dar a conocer cómo se va a delimitar el proyecto. Los criterios de inclusión comprenden los requerimientos para que este pueda funcionar, entre los cuales se encuentran la conexión a internet, una cámara para el monitoreo, una raspberry pi 4 que realizará el envío de imágenes desde la cámara hacia el servidor, un servidor que realice todo el procesamiento y en el cual se alojará la aplicación web junto con la base de datos, el conjunto de imágenes que se usan para el entrenamiento y una máquina en la que se pueda acceder a la aplicación web. Los criterios de exclusión se refieren a los aspectos que no se deben de tomar en cuenta para el desarrollo del proyecto o los aspectos que se debe de limitar el sistema para su implementación. Estos abarcan las razas de animales a las cuales se limita el estudio, las especies que se estudian y los estados que se predicen.

[7] https://developer.amazon.com/es-ES/alexa

2.4 Métricas de evaluación de la plataforma

A pesar de que el entorno para implementar la aplicación es completamente virtual, resulta necesario evaluar el desempeño del proceso completo de la predicción con el fin de conocer el comportamiento en la situación que se plantea y así estimar el rendimiento en un escenario real.

Las métricas que se consideran para definir el desempeño del sistema como bueno o malo son la latencia, la precisión de la predicción y el rendimiento del servidor.

- **Latencia(B/s):** si esta métrica es alta puede significar que se tengan problemas en la conexión al realizar el envío de la imagen desde la cámara hacia el servidor o al realizar consultas a la base de datos.

- **Precisión de la Predicción(%):** Sirve para definir que tan acertada fue la predicción realizada. Al inicio puede resultar baja debido a que el modelo no cuenta con un dataset muy amplio que permita realizar un entrenamiento con menos errores, sin embargo, a medida que el modelo se entrena con más imágenes, se puede mejorar dicha métrica para que tenga un porcentaje de error más bajo.

- **Rendimiento del Servidor:** mientras mejor sea el rendimiento, mejor será el nivel de satisfacción del usuario, este se mejora al administrar de manera eficiente la cantidad de solicitudes servidas frente al número de solicitudes entrantes. La ram consumida al ejecutar acciones de predicción, consulta e inserción de datos es la métrica usada para determinar el rendimiento del servidor, entre menor sea dicha métrica, se considera que se tiene una distribución efectiva de las tareas y de la carga sobre el servidor.

Para realizar las pruebas de estrés necesarias para evaluar el desempeño del sistema completo se hace uso de Apache JMeter [8] el cual automatiza las pruebas de comportamiento y desempeño.

También es importante medir la eficacia. Para esto se usa el nivel de satisfacción del usuario pues ayuda a determinar que tan acertada fue la predicción y el rendimiento del sistema durante el proceso. La siguiente fórmula será de ayuda para determinar el nivel

[8] https://jmeter.apache.org

de eficacia.

$$Eficacia = \frac{RA}{RP} x100 \tag{2.1}$$

- *RA*: resultados alcanzados/porcentaje de acierto dado por el usuario.
- *RP*: resultados previstos/precisión dada por la predicción.

2.5 Métricas de satisfacción del usuario.

Se tiene una encuesta que pretende determinar la experiencia que el usuario podría tener con el sistema y la utilidad que otorgaría a este una vez implementado. Los usuarios incluyen a los veterinarios, encargados de refugios de animales y dueños de mascotas quienes consideran muy importante el bienestar de los animales. En Guayaquil se encuentran alrededor de 30 refugios de animales con un promedio de 12 encargados cada uno, más de 40 clínicas veterinarias que tienen en promedio 4 miembros. Debido a que la cantidad de personas que son dueñas es muy alta no se puede determinar un número específico que englobe a dicha población. De momento se trabajará con un tamaño de población de 520 individuos, entre veterinarios, encargados de refugios, y ayudantes de clínicas veterinarias. Con está cantidad como tamaño de la población se aplica la siguiente fórmula para conocer el tamaño de la muestra

$$muestra = \frac{\frac{z^2 p(1-p)}{e^2}}{1 + \frac{z^2 p(1-p)}{e^2 N}} \tag{2.2}$$

Donde:

- *N*: tamaño de la población
- *e*: margen de error %
- *p*: probabilidad a favor %
- *z*: puntuación z; representa la cantidad de desviación estándar que las proporciones se alejan de la media.

Tabla 2.1: Nivel de confianza relacionada a la puntuación z

Nivel de confianza deseado	Puntuación z
80%	1,28
85%	1,44
90%	1,65
95%	1,96
99%	2,58

Considerando la población de 520 individuos, se ha decidido trabajar con un margen de error del 10% y un nivel de confianza del 80% es decir, puntuación z de 1,28. Desarrollando la fórmula, se obtiene una muestra de alrededor de 38 individuos.

Las respuestas de la encuesta representan qué tan de acuerdo o en desacuerdo se encuentran los usuarios con respecto a la pregunta que se plantea como se puede ver en la siguiente tabla.

Tabla 2.2: Respuesta de la encuesta

Puntuación	Significado
5	Muy de acuerdo
4	De acuerdo
3	Neutral
2	En desacuerdo
1	Muy en desacuerdo

Para obtener el resultado de esta métrica, se calcula el promedio el cual se traduce en la satisfacción media del usuarios respecto al sistema. El resultado también puede apoyarse en un porcentaje el cual determinará la satisfacción neta de los doctores usando la siguiente fórmula:

$$SN = \frac{DTA}{TD} x100 \tag{2.3}$$

- *SN*: Satisfacción neta.
- *DTA*: : Cantidad total de doctores que calificaron con 4 y 5.
- *TD*: Total de doctores participantes.

Las preguntas realizadas servirán para realizar un análisis general tomando como base todas las respuestas e individual, es decir, por cada pregunta, de esta manera

se podrán generar gráficas y obtener estadísticas para conocer las reacciones frente al sistema y la factibilidad que tendrá el proyecto.

CAPÍTULO 3

3. Diseño e Implementación

La implementación del proyecto permite determinar qué tan factible sería en situaciones reales, junto con esta implementación, se debe documentar cada paso y sección desarrollada, de manera que se conozcan todas las especificaciones técnicas, razón de uso de tecnologías y estructuras de las mismas. En este capítulo se encuentra dicha documentación junto con diagramas de escenarios de uso del proyecto. Se empieza relatando sobre la construcción del modelo de aprendizaje automático, luego se habla sobre el desarrollo de la aplicación web junto con el middleware, se procede a explicar la estructura de la base de datos y cómo esta se enlaza con el marco de trabajo de desarrollo web escogido, finalmente, se construyeron diagramas que permiten entender las interacciones de los usuarios con el sistema y las interacciones internas del sistema para funcionar de forma correcta.

3.1 Modelo de Aprendizaje Automático

El modelo de aprendizaje representa una pieza fundamental en el desarrollo del proyecto. La función del modelo es predecir el estado de ánimo del animal al momento de capturar la imagen y enviarla hacia el servidor. A continuación se explica todos los componentes que hacen parte de este modelo.

3.1.1 Conjunto de datos para el entrenamiento

El conjunto de datos usado para entrenar el modelo se obtuvo de varias fuentes, los repositorios *https://images.cv*[1] y *Stanford Dogs*[2] sirvieron como punto inicial para obtener

[1] https://images.cv
[2] http://vision.stanford.edu/aditya86/ImageNetDogs/

las imágenes usadas en el primer entrenamiento. Para extender el conjunto de datos, se tomaron fotografías desde las imágenes de *Google*, posterior a esto, también se solicitó a dueños de mascotas cuyas razas pertenecen al objeto de estudio para que estas sean cedidas y tomar fotografías de estas en diferentes estados emocionales. La figura 3.1 muestra un grupo de fotografías de la raza *golden retriever* pertenecientes al estado *feliz* que se han usado para el entrenamiento.

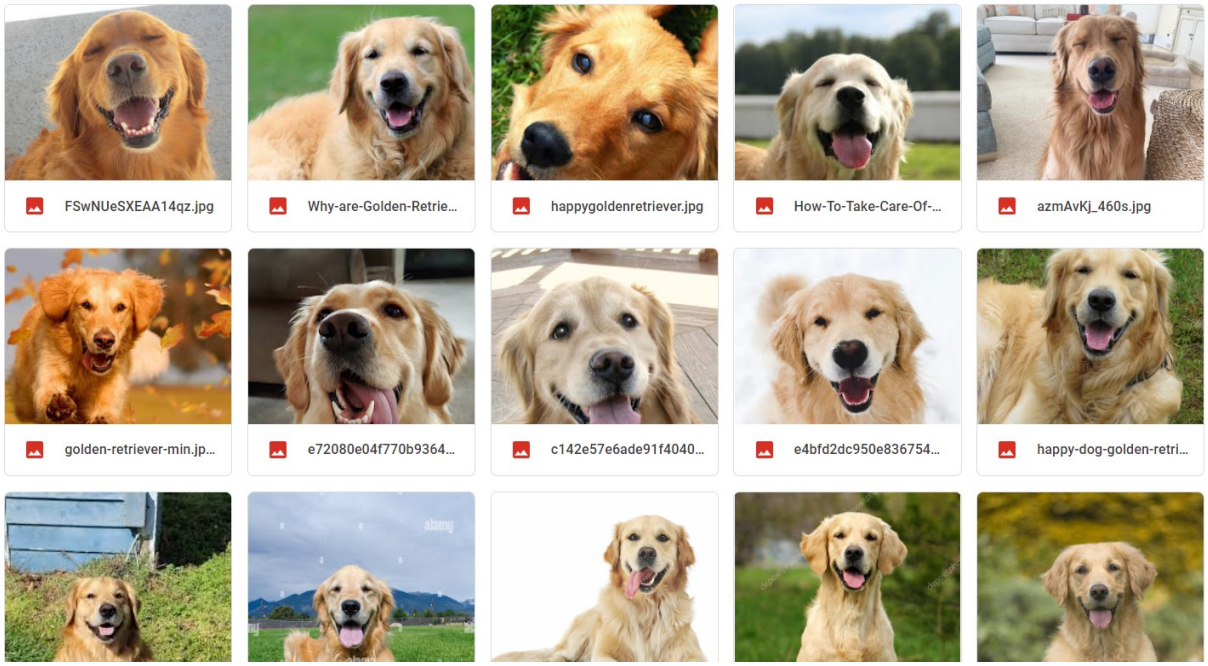


Figura 3.1: Imágenes del repositorio usadas en el entrenamiento

3.1.2 Diseño del Algoritmo

El modelo construido pertenece al grupo de algoritmos de aprendizaje supervisado. Esto quiere decir que se recibe un conjunto de datos del operador el cual incluye las entradas y salidas esperadas. En este proyecto el modelo recibe las imágenes respectivamente etiquetadas para poder entrenarse y de esa manera generar las salidas esperadas, las cuales corresponden a los estados de los animales. Adicionalmente, se puede clasificar al algoritmo como una red neuronal, es decir que comprende una serie de capas, las cuales se conectan entre sí para poder realizar las predicciones requeridas.

En la figura 3.2 se puede observar el funcionamiento básico de la red neuronal en este proyecto. Se recibe como entrada la imagen del animal, esta es analizada y procesada por la red neuronal, al detectar patrones similares con la data de entrenamiento se puede

generar una salida con el estado detectado en la fotografía.

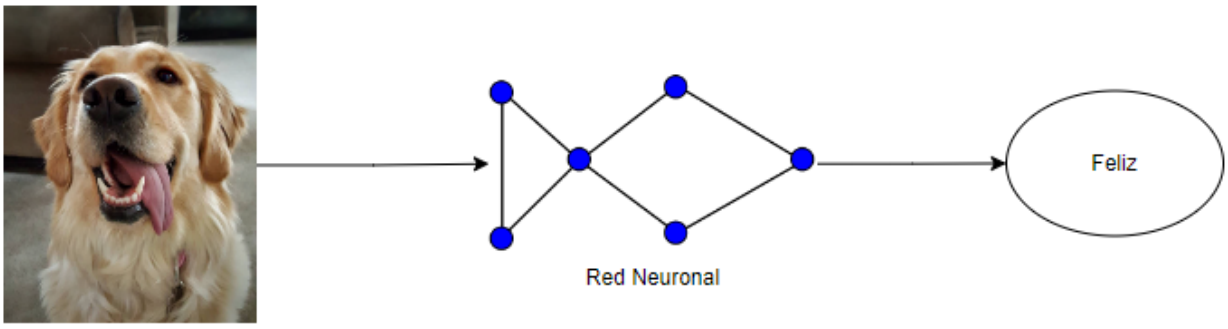


Figura 3.2: Funcionamiento de Algoritmo

El algoritmo utiliza las siguientes librerías para su funcionamiento: *numpy*, *pandas*, *tensorflow*, *seaborn*, *keras*, *opencv*, *random*, *matplot*, *sklearn* y *os*.

El código en 3.1 se encarga de cargar la información para entrenar al modelo, las imágenes son reescaladas para ser estandarizadas y adaptadas a las dimensiones requeridas. En la variable *labels* se almacenan las etiquetas asignadas a las imágenes, *img_size* representa la dimensión a la que se va a reescalar la imagen. Dentro de la función *get_data* se van a recorrer las carpetas en donde se contienen las imágenes (es importante que los nombres de las carpetas sean los mismos que las etiquetas). Luego se leen las imágenes y finalmente se las añade a un arreglo en donde se las va a almacenar para realizar el entrenamiento. En caso de que se dé algún error este se va a imprimir en la consola.

Código 3.1: Carga de Información del Algoritmo de Aprendizaje

```
// Carga de informacion
#Asignacion de etiquetas
labels = ['happy', 'sad', 'hungry', 'unknown']
#Tamaño de imágenes para ser redimensionadas
img_size = 150
#Funcion para carga de informacion
def get_data(data_dir):
    data = []
    for label in labels:
        path = os.path.join(data_dir, label)
        class_num = labels.index(label)
```

```
            errors = 0
            for img in os.listdir(path):
                try:
                    img_arr = cv2.imread(os.path.join(path, img), cv2.IMREAD_COLOR)
                    resized_arr = cv2.resize(img_arr, (img_size, img_size))
                    data.append([resized_arr, class_num])
                except Exception as e:
                    errors+=1
                    print(img,e)
            print(label, errors)
    return np.array(data)
#obtencion de imagenes
data = get_data("./golden_retriever/golden_retriever")

x = []
y = []

for feature, label in data:
    x.append(feature)
    y.append(label)

x = np.array(x) / 255
x = x.reshape(-1, img_size, img_size, 3)
y = np.array(y)
from sklearn.preprocessing import LabelBinarizer

#binarizacion de etiquetas
label_binarizer = LabelBinarizer()
y = label_binarizer.fit_transform(y)
x_train,x_test,y_train,y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)
del x,y
```

En el fragmento de código en 3.2, se entrena al modelo. Se pasa la data por varias iteraciones o *epochs* para mejorar la precisión del algoritmo. Se usa VGG19, la cual es una red convolucional preentrenada de 19 capas, esto resulta útil debido a que puede detectar características y patrones con mayor facilidad en las imágenes.

La función de activación es *softmax*, esta se escogió porque se desea que se represente una distribución de probabilidad sobre una variable discreta con varios valores.

Se usa *adam* como optimizador debido a que tiene eficiencia computacional, bajos requerimientos de memoria, es invariante al reescalamiento diagonal de gradientes, y funciona de manera óptima en casos en los que se tiene fuentes de datos amplias.

La pérdida usada es *sparse_categorical_crossentropy* debido a que se tiene más de dos etiquetas. La métrica que se asignó es *accuracy*, esta métrica crea las variables de *total* y *count*, las cuales se usan para comparar la frecuencia en que *y_pred* coincide con *y_true*. *y_pred* representa la data predecida por el modelo, mientras que *y_true* es la data que se envía para que el modelo se ajuste.

Código 3.2: Entrenamiento del Algoritmo de Aprendizaje

```
// Entrenamiento del algoritmo
#carga de red neuronal preentrenada
pre_trained_model = VGG19(input_shape=(150,150,3), include_top=False, weights="imagenet")

for layer in pre_trained_model.layers[:19]:
    layer.trainable = False
#Creacion de modelo
model = Sequential([
    pre_trained_model,
    MaxPool2D((2,2) , strides = 2),
    Flatten() ,
    Dense(2 , activation='softmax')])
#Compilación de modelo
model.compile(optimizer = "adam" , loss = 'sparse_categorical_crossentropy' , metrics = ['accuracy'])
model.summary()
del pre_trained_model
```

El algoritmo descrito en el código 3.3 corresponde a la función usada para predecir el estado en una imagen, esta recibe la ruta de la imagen, luego es reescalada a 150x150px, y finalmente se genera una respuesta sobre el estado detectado.

Código 3.3: Funcion para predecir el estado en una imagen

```
// Validacion del modelo
def validate_model(path_img):
  img = tf.keras.utils.load_img(
      path_img,
      target_size=(150, 150)
  )
  img_array = tf.keras.utils.img_to_array(img)
  img_array = tf.expand_dims(img_array, 0) # Create a batch

  predictions = model.predict(img_array)
  print(predictions)

  id_prediction = np.argmax(predictions)
  prediction = np.amax(predictions, axis=1)

  plt.imshow(cv2.imread(path_img))


  print({"message":"This boy might be feeling {}".format(labels[id_prediction]),
         "precision": "{:.4f}%".format(100 * prediction[0])}
  )
```

3.2 Transmisión de Video mediante Raspberry Pi

El mini-computador *Raspberry Pi* es un componente clave en este proyecto. Con la capacidad de conectar una cámara de vídeo, se pueden transmitir imágenes y vídeos en tiempo real a través de la plataforma web desarrollada. Esto permitirá una experiencia

única al ver las transmisiones en vivo de cada mascota registrada en la plataforma, aumentando la interacción y conexión con ellas.

3.2.1 Configuración del Servidor de Video

El servidor de video en la Raspberry Pi es parte fundamental para la transmisión en tiempo real de las mascotas en la plataforma web. Para ello, se utiliza el paquete de software de código abierto *motion* [3].

Después de la instalación, se deben editar los ajustes importantes en el archivo de configuración en */etc/motion/motion.conf*, tales como la resolución de la cámara, la ruta de guardado de las imágenes y la tasa de cuadros por segundo. Una vez en ejecución, se puede acceder a la transmisión en tiempo real mediante un dispositivo remoto ingresando la dirección IP de la Raspberry Pi seguida del puerto 8080.

3.2.2 Predicción de Emociones Mediante el Modelo Predictivo

La realización de predicciones de emociones en las mascotas es un proceso que se lleva a cabo mediante la conexión de una cámara a una Raspberry Pi y la realización de peticiones *HTTP POST*. Con un script 3.4 en *Python* [4], se toman fotografías periódicas de la mascota y se envían a un servidor para su análisis y predicción del estado emocional. Estas predicciones son realizadas cada minuto y proporcionan una evaluación precisa para asegurar el bienestar de la mascota.

Código 3.4: Script Periódico Para Realizar Predicciones Desde Raspberry Pi

```
import requests
import time

while True:
  # Obtener imagen desde servidor local
  image = requests.get('http :// localhost:8081/0/stream')
```

[3] https://motion-project.github.io
[4] https://www.python.org

```
# Enviar imagen en petición HTTP POST a servidor externo
response = requests.post(
    'https :// goodboy-pwza.onrender.com/api/predict',
    data={'image': image, 'animal_type': 'dog'}
)

# Obtener respuesta en formato JSON
result  = response.json()

# Procesar resultado
print ('Estado del animal:',  result ['state'])

# Esperar 1 minuto antes de repetir  la  tarea
time.sleep(60)
```

Adicionalmente, el servidor principal de la aplicación se encargará de almacenar las respuestas en una base de datos, para poder tener un registro de estos resultados y que a su vez puedan ser compartidos con la plataforma web.

3.3 Diseño y Programación de la Aplicación Web

3.3.1 Programación del Middleware

Django REST Framework es un marco de trabajo que facilita el desarrollo de APIs bajo la arquitectura REST. Incluye gran cantidad de código para reutilizar entre los que se encuentran Views, Resources, funciones, etc. Y una interfaz administrativa desde la que se pueden realizar pruebas sobre las operaciones HTTP POST y GET. Es por esto que se eligió esta tecnología para el desarrollo del *middleware*. Django hace uso de clases genéricas, estas se desarrollaron como un atajo para los patrones de uso común [5] La tabla 3.1 explica las expecifiaciones técnicas del *middleware* construido con DJango. b

En el código 3.5 se observa como se obtiene un campo $'tipo\_animal'$ desde una petición para devolver una respuesta de un *queryset* serializado como *json* que contiene

[5]https://www.django-rest-framework.org/api-guide/generic-views/

Tabla 3.1: Especificaciones técnicas del middleware

Version	4.1.3
Puerto	8000
Dominio	https://goodboy-pwza.onrender.com/api/

a todos los animales registrados en la base de datos que corresponden a la especie que se busca.

Código 3.5: Código para devolver Animales

```
class Animals(APIView):
    def get(self, request):
        if "tipo_animal" in request.query_params:
            queryset = Animal.objects.filter(
                species_common=request.query_params["tipo_animal"]
            )
        else:
            queryset = Animal.objects.all()

        serializer = AnimalSerializer(queryset, many=True)

        return Response(serializer.data)
```

El bloque de código 3.6 demuestra el código del view que hace una inserción en la base de datos de una predicción realizada por el modelo de Aprendizaje Automático y enviada por un método POST a la ruta de la API. Los campos que se obtienen son la ruta de la imagen donde se va a almacenar, el estado predecido, la precisión de la predicción dada por el modelo y el animal del cual se realizó la predicción, se guardará el registro con estos datos junto con el instante en el que se hace la inserción en formato de fecha y hora de Python.

Código 3.6: Código para Obtener Estados

```
class ReturnAnimalState(APIView):
    def post(self, request, *arg, **kargs):
```

```
image_path = request.POST.get('image_path',None)
state  = request.POST.get('predicted_state',None)
accuracy = request.POST.get('accuracy',None)
animal = request.POST.get('animal',None)
date = datetime.now()
new_state = StateHistory.objects.create(
    image_path = image_path,
    state  = state,
    accuracy = accuracy,
    animal = animal,
    date = date,
)
serialized_state  =  StateSerializer(new_state, many = False)

return  Response({
    serialized_state.data
})
```

3.3.2 Diseño de Interfaz de Usuario

Flutter es un marco de trabajo multiplataforma que facilita el desarrollo de aplicaciones móviles y web con una experiencia de usuario atractiva y animaciones fluidas. Incluye una amplia gama de herramientas y widgets para crear aplicaciones altamente personalizables. Es un framework de código abierto desarrollado por Google que permite desarrollar aplicaciones para iOS, Android y web con un solo código base. Además, Flutter hace uso de un lenguaje de programación propio llamado Dart, el cual facilita la creación de interfaces atractivas y animadas. Esta tecnología se ha elegido para el desarrollo de la aplicación debido a sus características intuitivas y versatilidad en el desarrollo de aplicaciones web.

En la figura 3.3 se puede observar la interfaz web en la sección de gatos. Se muestra un diseño intuitivo, en donde fácilmente se puede cambiar de sección o tipo de animal y seleccionar cualquier animal registrado para poder ver más información sobre él, incluyendo su transmisión en vivo. La interfaz web permite al usuario navegar de

manera sencilla y rápida a través de las distintas mascotas registradas en el sistema. Esta funcionalidad aumenta la eficiencia y la experiencia de usuario en la plataforma.

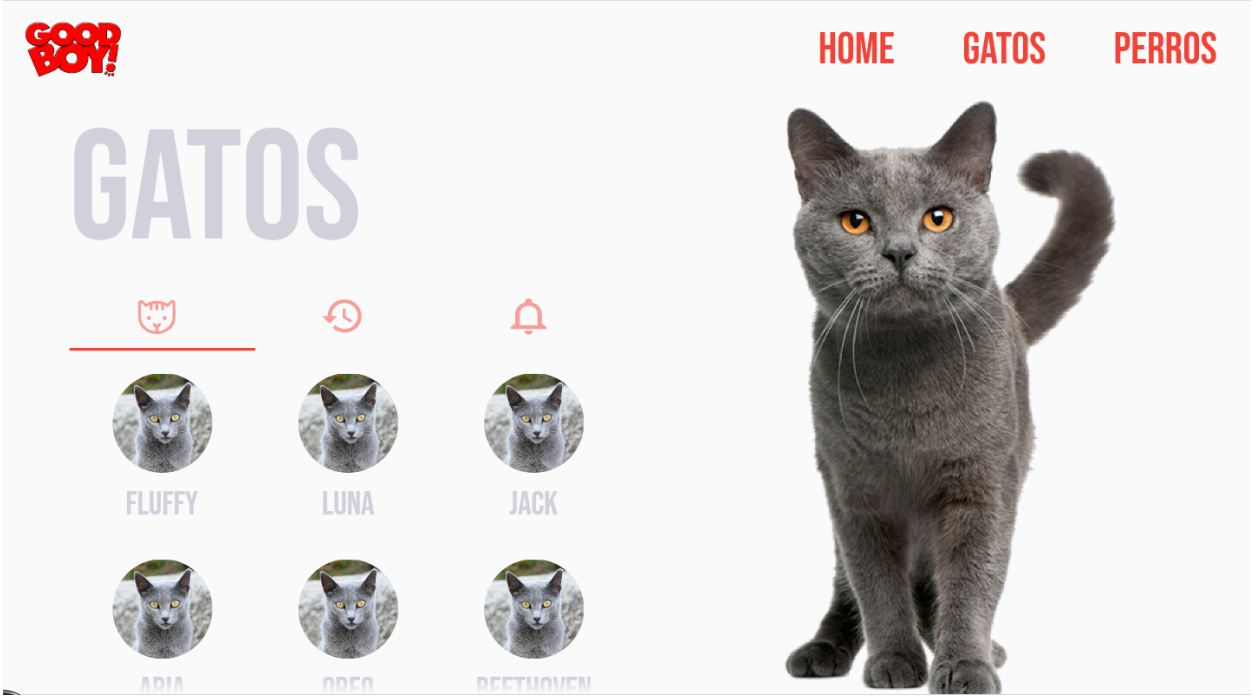


Figura 3.3: Interfaz de interacción con Animales

El bloque de código 3.7 muestra cómo se realiza la petición HTTP para obtener los animales desde el servidor que se aloja en Django. En el bloque de código 3.8 se puede ver cómo se renderiza la información de las mascotas en la web. Esta información incluye los datos de cada mascota, como su nombre e imagen. Este proceso permite al usuario ver la información sobre las mascotas, lo cual es una funcionalidad fundamental para la plataforma.

Código 3.7: Código para Obtener la Información de los Animales desde el Servidor

```
@override
void onInit() async {
  super.onInit();

  final animalResponse = await httpClient.get(
    "/api/animals/",
    query: {"tipo_animal": tipoAnimal == TipoAnimal.gato ? "cat" : "dog"},
  );
```

```
    final alarmsResponse = await httpClient.get(
      "/api/alarms/",
      query: {"tipo_animal": tipoAnimal == TipoAnimal.gato ? "cat" : "dog"},
    );

    _animales.addAll(
      json.decode(animalResponse.bodyString!),
    );
    _alarmas.addAll(
      json.decode(alarmsResponse.bodyString!),
    );

    _loading.value = false;
  }
```

Código 3.8: Código para Renderizar la Sección de Animales

```
Padding(
  padding: const EdgeInsets.only(bottom: 16),
  child: TabBar(
    tabs: [
      Tab(
          icon: AnimalOption(
        onAction: () {},
        svgPath: "assets/svg/${controller.tipoAnimal == TipoAnimal.gato ? 'cat_icon.svg' : 'dog_icon.svg'}",
      )),
      Tab(
          icon: AnimalOption(
        onAction: () {},
        svgPath: "assets/svg/history_icon.svg",
      )),
      Tab(
          icon: AnimalOption(
```

```
            onAction: () {},
            svgPath: "assets/svg/ notification_icon .svg",
          )),
        ]
            .map((option) => Padding(
                  padding: const EdgeInsets.only(right: 8),
                  child: option,
                ))
            .toList(),
      ),
    ),
    Flexible(
      child: Obx(
        () => (controller.loading)
            ? const Center(
                child: CupertinoActivityIndicator(
                  radius: 32,
                  color: Color(0xFFD2D0DB),
                ),
              )
            : FadingEdgeScrollView.fromScrollView(
                child: GridView.count(
                  controller: ScrollController(),
                  crossAxisCount: 3,
                  children: controller.animales
                      .map(
                        (a) => InkWell(
                          onTap: () {
                            Get.toNamed(
                              "${Get.currentRoute}/${a['id']}",
                              arguments: a["name"],
                            );
                          },
                          child: AnimalAvatar(
```

```
                    key: Key(a["id"].toString()),
                    url: null,
                    nombre: a["name"],
                    tipo: controller.tipoAnimal,
                  ),
                ),
              )
              .toList(),
          ),
        ),
    ),
)
```

3.4 Diseño de la Base de Datos

Postgresql es una base de datos relacional, en este proyecto ofrece facilidad de enlace con Django al momento de realizar la construcción de las tablas.

3.4.1 Especificaciones técnicas

La base de datos cumple con ciertas características para ser integrada en este proyecto, las cuales se pueden ver en la tabla 3.2:

Tabla 3.2: Especificaciones técnicas de la base de datos

Version	15.0
Puerto	5432
Dominio	dpg-ce6ms42rrk071o64qb2g-a.oregon-postgres.render.co

3.4.2 Interfaz de Comunicación con la Base de Datos

Django se usó también para la construcción de la base de datos. Gracias a que posee *Models*, es decir, fuentes de información únicas que contienen los campos y comportamientos de los datos que se almacenan. En otras palabras, un Model en Django representa una tabla en la base de datos. La figura 3.4 es un ejemplo de un *Model*, en

este caso, del Modelo Animal, el cual sirve para registrar a los animales del recinto. Como se puede ver, se tienen los campos, con las características respectivas, como la longitud, si estos son obligatorios o incluso si son *Char*, *Integers*, *Float*, *Datetime*, etc.

```
class Animal(models.Model):
    name = models.CharField(max_length = 180, blank = False, null = False)
    species_common = models.CharField(max_length = 180, choices = ANIMAL_CHOICES, blank = False, null = False)
    species_scientific = models.CharField(max_length = 180, blank = True, null = True)
    race = models.CharField(max_length = 180, choices = RACE_CHOICES, blank = True, null = True)
    gender = models.CharField(max_length = 180, choices = GENDER_CHOICES, blank = True, null = True)
    age = models.IntegerField(blank = True, null = True)
    image = models.FileField(upload_to=get_image_profile, blank=True, null=True)
```

Figura 3.4: Ejemplo de modelo para Animal

3.4.3 Arquitectura de la Base de Datos

En la imagen 3.5 se observa la arquitectura de la base de datos. El ERD fue generado por medio de la herramienta de ERD generator de *PGadmin*[6]. Como se puede ver, hay tablas que no corresponden a las que intervienen en el proyecto como tal, por ejemplo, *auth_group_permissions*, *auth_user_user_permissions*, *auth_user_groups*, entre otras, estas tablas son generadas automáticamente por django para que el proyecto pueda funcionar normalmente.

- La tabla *auth_group* corresponde a los grupos de usuario que se crean para poder organizar a los usuarios de acuerdo a permisos de grupo.
- *auth_user* tiene un registro de todos los usuarios creados, almacena información correspondiente al nombre de usuario, contraseña primer nombre, apellido, etc.
- *auth_user_groups* contiene la relación entre la tabla *auth_user* y *auth_group*.
- En *auth_permission* se guardan los permisos que se crean, estos pueden determinar a qué módulos de la aplicación se puede acceder, la información que se puede visualizar, las acciones que se pueden ejecutar, etc.
- *auth_group_permissions* relaciona a los elementos de la tabla *auth_permission* y *auth_group*, de esta manera se puede saber los permisos que posee cada grupo.
- *auth_user_user_permissions* relaciona a la tabla *auth_user* con *auth_permission*, esto con el fin de asignar permisos de manera idividual a cada usuario.

[6]https://www.pgadmin.org/

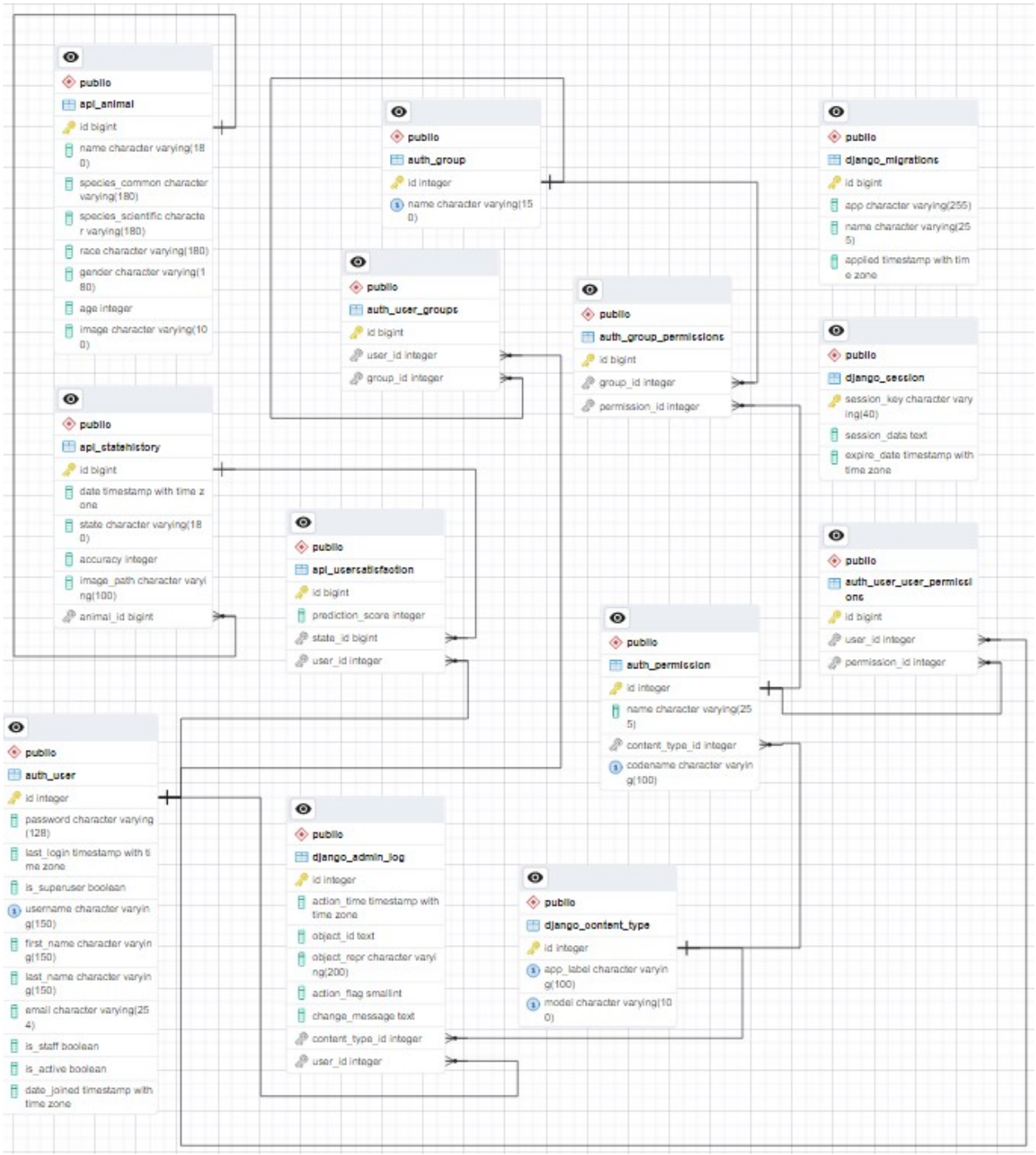


Figura 3.5: Diagrama de Entidad Relación de la Base de Datos

- *django_content_type* contiene a las acciones que se pueden ejecutar en la aplicación, de esta manera, la tabla *django_admin_log* puede tener un registro de las acciones ejecutadas y relacionarlas con cada usuario junto con la fecha y hora en que se ejecutó la acción.
- *django_migrations* almacena las migraciones ejecutadas en la base de datos, una migración se puede referir a cambios en una tabla, tablas creadas o eliminadas o cualquier manipulación de la estructura de la base de datos.
- Cuando se esté usando el backend de la base de datos, se creará un registro y este se almacenará en la tabla *django_session* hasta que el usuario cierre su sesión de forma manual, luego de esto, el registro se elimina. Si la sesión expira, este registro no se eliminará automáticamente.
- *api_statehistory* almacena los estados predecidos por el modelo junto con la fecha y hora respectiva, la ruta de la imagen, la precisión de la predicción y el animal del cual se realizó la predicción.
- La tabla *api_animal* almacena un registro de todos los animales que se encuentran en el recinto, la información que se almacena es el nombre del animal, el nombre común de especie a la que pertenece, el nombre científico de la especie, la raza, la edad y una imagen de dicho animal.
- *api_usersatisfaction* almacena el nivel de satisfacción del usuario con respecto a la predicción ejecutada, en esta tabla se guardan datos como el puntaje de la predicción, el usuario que dio el puntaje y la predicción que se realiza.

3.5 Integración con Alexa

Alexa es un asistente virtual inteligente desarrollado por Amazon, que permite a los usuarios interactuar con dispositivos como Echo Dot mediante comandos de voz. Esta plataforma se integra con una amplia variedad de servicios y aplicaciones, permitiendo una interacción fluida y personalizada.

En este proyecto, Alexa se utiliza para notificar a las personas el estado de ánimo de sus mascotas en tiempo real. Para lograr esto, se utiliza la plataforma Voice Monkey, la

cual permite la integración con Alexa mediante triggers virtuales. Voice Monkey brinda la capacidad de ejecutar una rutina en Alexa, mediante una petición HTTP GET, lo que permite la transmisión de información desde la plataforma de monitoreo de animales a Alexa.

3.5.1 Uso de Voice Monkey

Voice Monkey es una plataforma que permite integrar aplicaciones con Alexa, permitiendo ejecutar una rutina de Alexa a través de una petición realizada desde una página web, aplicación móvil o cualquier tecnología que permita realizar peticiones HTTP. Gracias a esta funcionalidad, se ha podido realizar la integración con Alexa para notificar el estado de ánimo de las mascotas, lo cual es una funcionalidad clave para el cuidado y monitoreo remoto de estos animales.

3.6 Interacción de usuarios con el sistema

El proceso de interacción del usuario con el sistema consta de dos etapas bien diferenciadas. En primer lugar, se tiene el proceso en donde el usuario revisa la página y accede a información correspondiente a los animales y sus estados. Este proceso se representa gráficamente en la figura 3.6, el primer paso es acceder por medio de la PC a la página la cual consulta a la nube los registros en la base de datos, paralelamente, la cámara se encuentra registrando a los animales y transmitiendo al servidor las imágenes para realizar la predicción correspondiente.

Luego de que se realiza el envío de la predicción se mostrará una notificación en la página con la nueva predicción, el usuario deberá evaluar dicha predicción y dar un puntaje que servirá para evaluar la precisión del modelo y determinar si se debe realizar un entrenamiento nuevamente con el fin de mejorar esta métrica. El proceso descrito se encuentra representado de manera gráfica en la figura 3.7

En la figura 3.8 se puede observar la explicación junto con capturas de la página, cómo es la secuencia de los módulos para acceder a la gráfica generada con los estados predecidos. El primer paso es acceder al módulo *home* desde el que se puede ingresar a los listados de gatos y perros. Una vez en el listado se puede seleccionar a uno de los

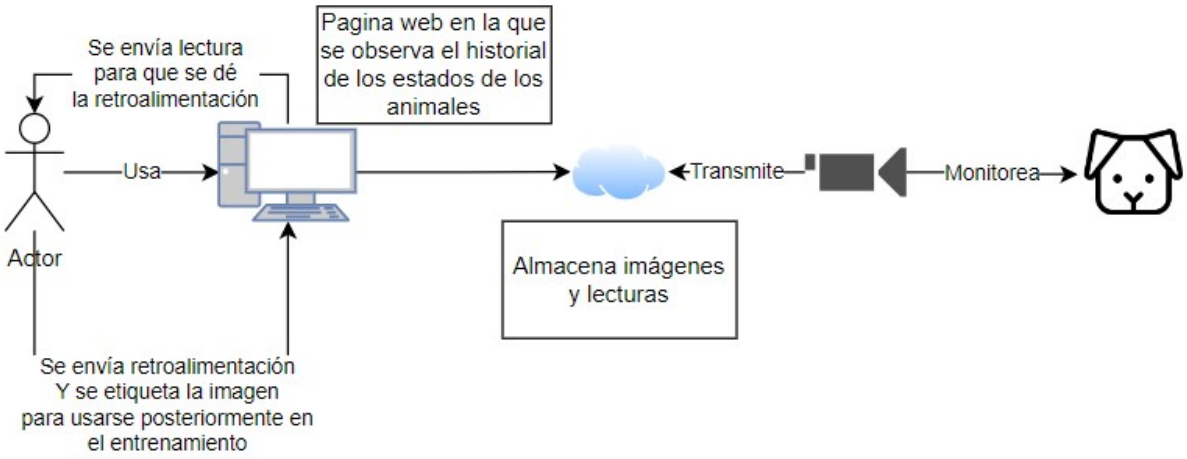


Figura 3.6: Interacción de usuario con el Sistema

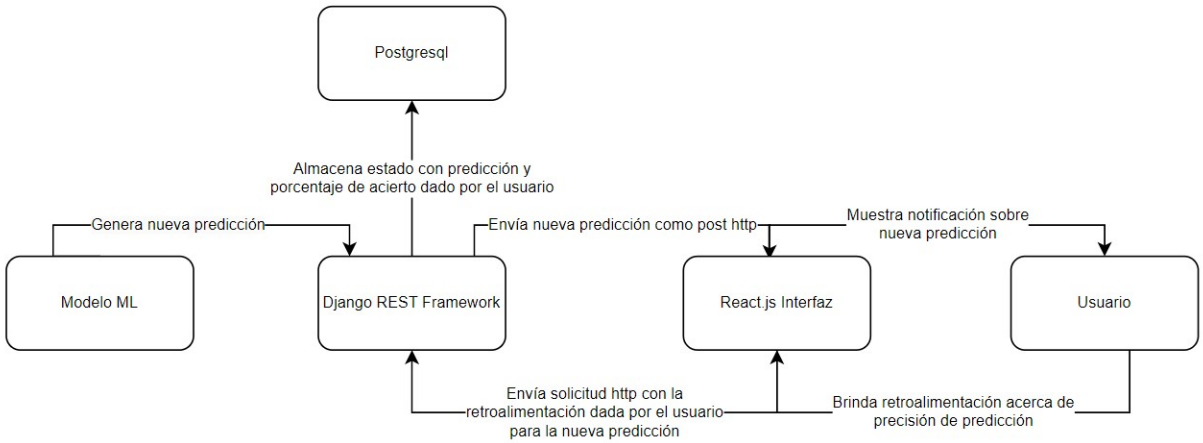


Figura 3.7: Retroalimentación de Usuarios

sujetos para poder ingresar a ver el gráfico generado con los estados de dicho animal ordenados por día.

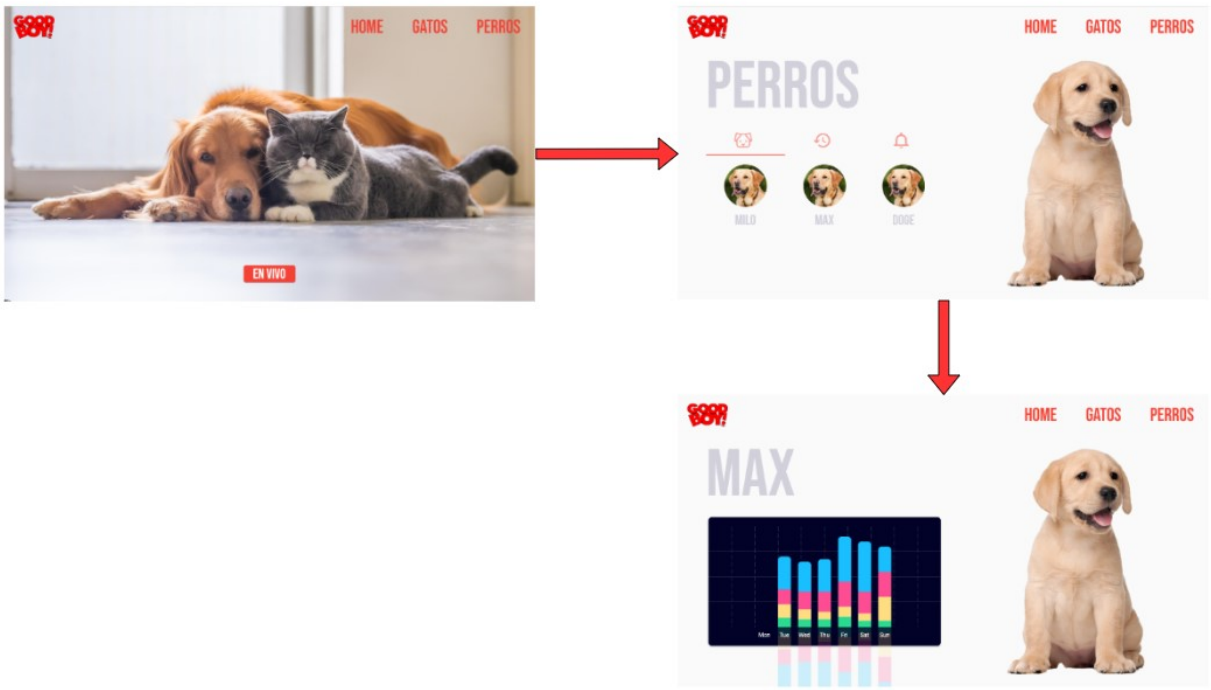


Figura 3.8: Secuencia de Interacción con la Página

3.7 Interacción de PC con la nube

Cuando la *PC* hace la consulta a la nube con el fin de obtener información y desplegarla se siguen una serie de pasos. Inicialmente, y con el fin de que la comunicación pueda llevarse a cabo, es necesaria una conexión a internet. Cuando se hace la consulta se accede a la aplicación web de *Django*, llegado a este punto, se realizan dos procesos simultáneos, se ejecuta una consulta al servidor y otra a la base de datos, esto devolverá la información de la predicción desde la tabla *StateHistory* y la imagen de la predicción, finalmente, se muestra la predicción y la imagen para ser mostrada en la página, se puede tomar la figura 3.9 como referencia visual para entender este proceso.

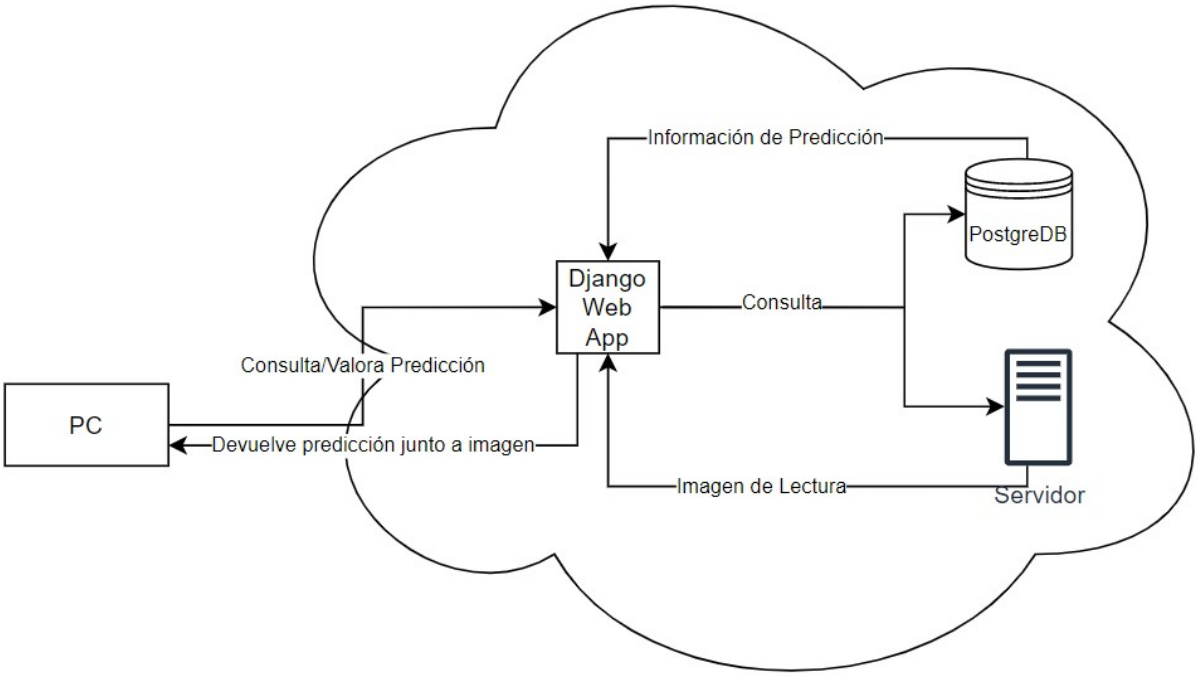


Figura 3.9: Interacción de PC con la nube

3.8 Presentación de Información

Los bloques que interactúan entre sí, siguen una secuencia para consultar y enviar respuestas con datos con el fin de mostrar la información. En el caso de la presentación de los animales 3.10, *Flutter* envía una solicitud *http* al *middleware* construido con *Django REST Framework*, el *middleware* obtiene la información desde el método *get* consulta a la tabla Animal y se devuelve el objeto de dicha tabla, finalmente el *API* devuelve la respuesta *HTTP* al *frontend* con los objetos serializados de la tabla consultada.

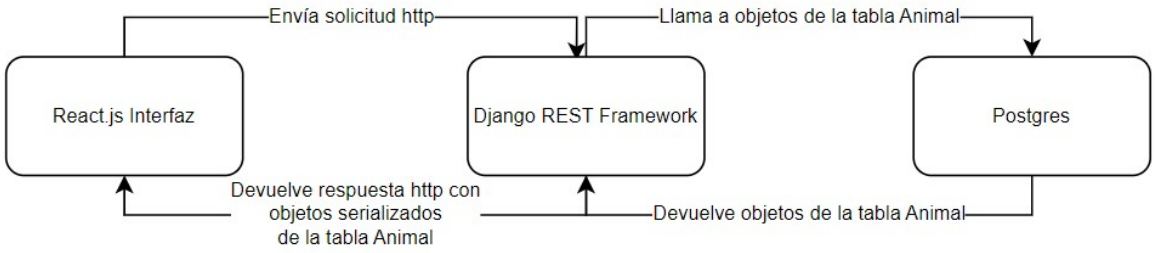


Figura 3.10: Presentación de Animales en el módulo del listado de animales

Para presentar los estados se sigue un proceso similar. El proceso se explica gráficamente en la figura 3.11, en primer lugar, el frontend envía la solicitud *HTTP* al *API* junto con el identificador del animal del cual se quieren obtener las predicciones, a su vez, el middleware consulta a la tabla *StateHistory* y se devuelve el objeto de dicha tabla, finalmente se envía la respuesta *HTTP* a *Flutter* y se envían los objetos serializados en formato *JSON* para ser mostrados en un gráfico.

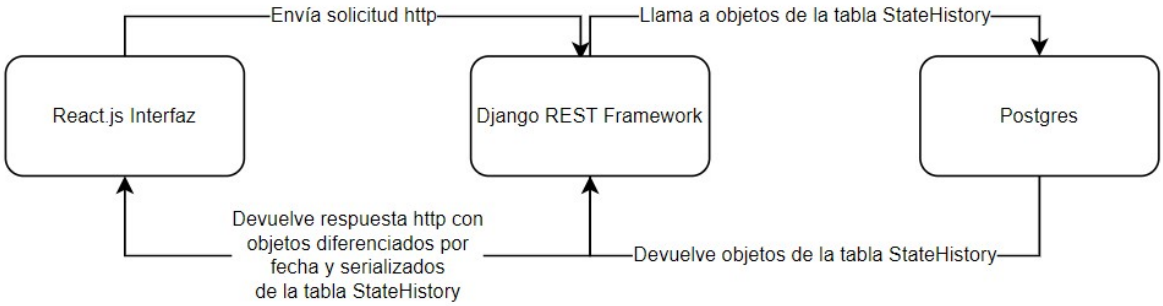


Figura 3.11: Presentación de Estados

CAPÍTULO 4

4. Análisis de Resultados

En este capítulo se presentan los resultados obtenidos de las pruebas realizadas en el sistema. Estas pruebas conllevan la carga que puede soportar el sistema, las pruebas y el entrenamiento realizado con el modelo de aprendizaje automático, el tiempo que toma en enviar una solicitud y obtener una respuesta. Además, a partir de las encuestadas realizadas a los clientes potenciales, se determina qué tan útil sería para veterinarios y cuidadores de refugios de animales la herramienta que se propone.

4.1 Escenarios de Prueba

Con el objetivo de medir la carga máxima simultánea que soporta el sistema se prepararon varios escenarios en los que se realizaron pruebas de envío de solicitudes http, empezando con 1, luego 100, luego 500 y por último 600. Se podrán notar cambios sorprendentes en la pérdida y el envío exitoso de paquetes. Se hizo un enfoque especial en 4 variables distintas, las cuales son:

- **Latencia Media:** Representa el tiempo que le toma al cliente en enviar un paquete y recibir una respuesta de dicho envío.
- **Rendimiento:** Es el desempeño del sistema ante una carga generada en el mismo.
- **Datos Enviados:** Representa la cantidad de información que se envía de manera exitosa hacia el servidor.
- **Datos Recibidos:** Son los datos recibidos por parte del servidor luego de realizar alguna petición.

Para poder ejecutar las pruebas de estrés en el sistema se usó el software *Apache JMeter*[1] el cual genera hilos para poder realizar envíos simultáneos de peticiones a un destino específico.

A continuación se describe el proceso en los 4 escenarios y cómo cambiaron las métricas en cada escenario.

4.1.1 1 solicitud http

Como se esperaba, una sola solicitud no representaba mayor desafío para el servidor. Al enviarse 1 petición no hay pérdida de datos y la latencia es baja. Al hacer la comparación entre los módulos a los cuales se realiza las peticiones se tienen los resultados de la tabla4.1 en donde la latencia y el volumen de datos recibidos son los más altos, lo cual se debe a que este es el módulo que más información contiene. Al hacer la petición a la página principal se obtuvo una latencia de 490ms y la cantidad de datos recibidos es la más baja debido a que la cantidad de información que se recibe no es muy alta. Al ejecutar la consultar a la API de animales tampoco se recibe un alto volumen de información debido a que lo que se devuelve solo comprende información de los animales y estos son pocos, por esto también la latencia es baja. Adicionalmente, se observa que el rendimiento es considerable, el sistema funciona bastante bien, es lo que se esperaba con 1 solicitud.

Tabla 4.1: Resultados de la prueba con 1 solicitud

Métrica	**Página de Inicio**	**API de Alarmas**	**API de Animales**
Latencia (ms)	490	497	428
Rendimiento (b/ms)	6.43	14.33	7.15
Datos Enviados (b/ms)	0.51	0.84	0.99
Datos Recibidos (b/ms)	5.92	13.49	6.16

4.1.2 100 solicitudes http

En la tabla 4.2 se pueden ver los resultados al ejecutar la consulta con 100 peticiones http simultáneas. Se observa que la latencia media disminuye en la página de inicio, sin embargo, en la API de alarmas aumentó en casi un 50% mientras que en la API de animales esta disminuyó a casi la mitad. Los datos recibidos varían también en la

[1] https://jmeter.apache.org

página de inicio, estos aumentan, al igual que para la API de animales, mientras que para la API de alarmas disminuyen. Se observa que a medida que las tasas de datos recibidos y enviados aumentan, también lo hace el rendimiento, lo cual denota que el sistema funciona bien y escala de manera eficiente para la peticiones que se ejecutan en él.

Tabla 4.2: Resultados de la prueba con 100 solicitudes

Métrica	**Página de Inicio**	**API de Alarmas**	**API de Animales**
Latencia Media (ms)	436.12	682.02	225.5
Rendimiento (b/ms)	7.04	10.44	13.57
Datos Enviados (b/ms)	0.57	0.62	1.88
Datos Recibidos (b/ms)	6.47	9.83	11.69

4.1.3 500 solicitudes http

Cuando se ejecutaron 500 solicitudes simultáneas la latencia para la página de inicio y la API de Alarmas empezó a aumentar considerablemente casi 10 veces más de lo que se tenía en el escenario de 1 solicitud, los datos recibidos disminuyeron, esto quiere decir que se perdió información en el proceso de respuesta por parte del servidor. Sin embargo, para la API de animales los parámetros se mantuvieron casi invariantes, probablemente se deba a que esto no representó gran carga para el servidor, por lo que pudo manejar las solicitudes con facilidad. Con 500 solicitudes se observa que el rendimiento disminuye considerablemente, al procesar tanta carga proveniente de las solicitudes realizadas es de esperar que el rendimiento del sistema disminuya.

Tabla 4.3: Resultados de la prueba con 500 solicitudes

Métrica	**Página de Inicio**	**API de Alarmas**	**API de Animales**
Latencia Media (ms)	4231.82	4745.49	225.59
Rendimiento (b/ms)	0.73	1.50	13.56
Datos Enviados (b/ms)	0.06	0.089	1.88
Datos Recibidos (b/ms)	0.69	1.41	11.69

En la figura 4.1 se observa la gráfica de la consulta ejecutada al módulo *Home* de la aplicación. Existe un pico, sin embargo, la mayor parte se mantiene por debajo de los 2400ms.

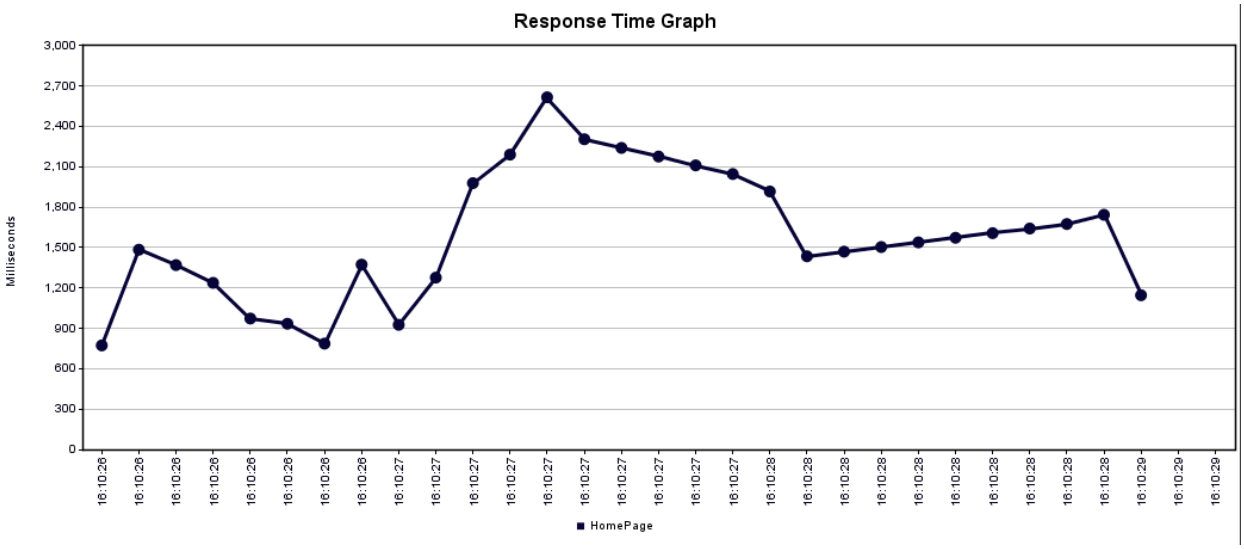


Figura 4.1: Tiempo de Respuesta de la página principal

Al ejecutar la consulta a la API de las alarmas se generó la gráfica que se observa en la figura 4.2, esta se muestra creciente en la mayor parte de su dominio. Es decir que el tiempo empleado para generar respuestas era cada vez mayor lo cual puede deberse a la sobrecarga que se generó en el servidor. Como ya se había explicado anteriormente, los datos generados por las alarmas son muchos, así que se razonable que exista sobrecarga en el servidor para poder entregar la respuesta con toda esta información mientras trata de procesar todas las solicitudes que le llegan.

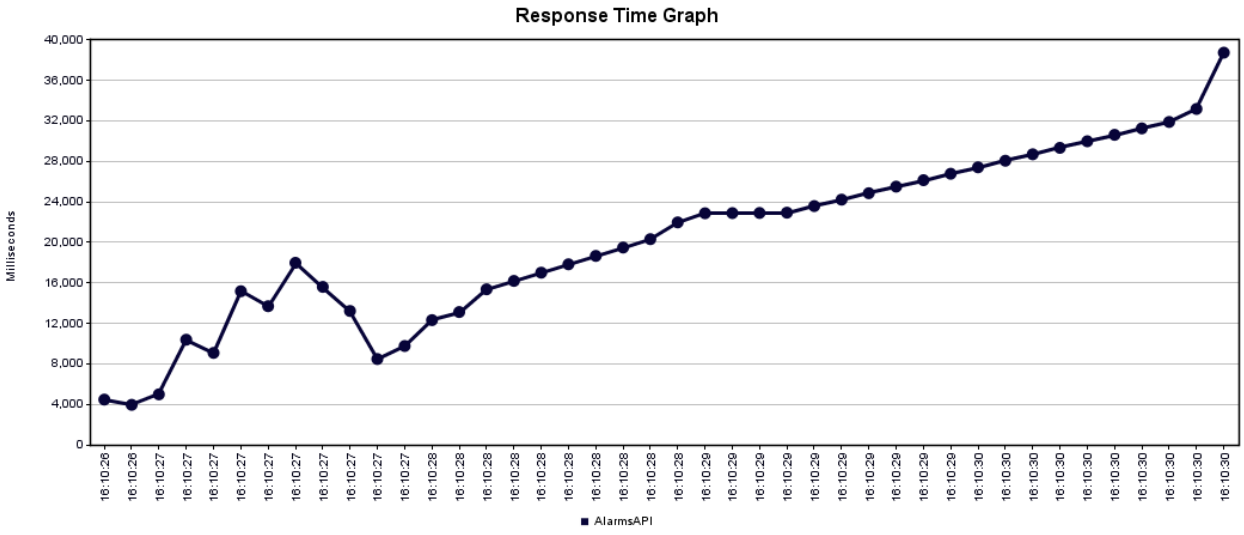


Figura 4.2: Tiempo de Respuesta de la API de Alarmas

El tiempo de respuesta de la API de animales es decreciente en la mayor parte de su dominio como se observa en la figura 4.3. Debido a que el volumen de estos datos no es grande, no le toma tanto tiempo generar la respuesta al cliente con toda esta información.

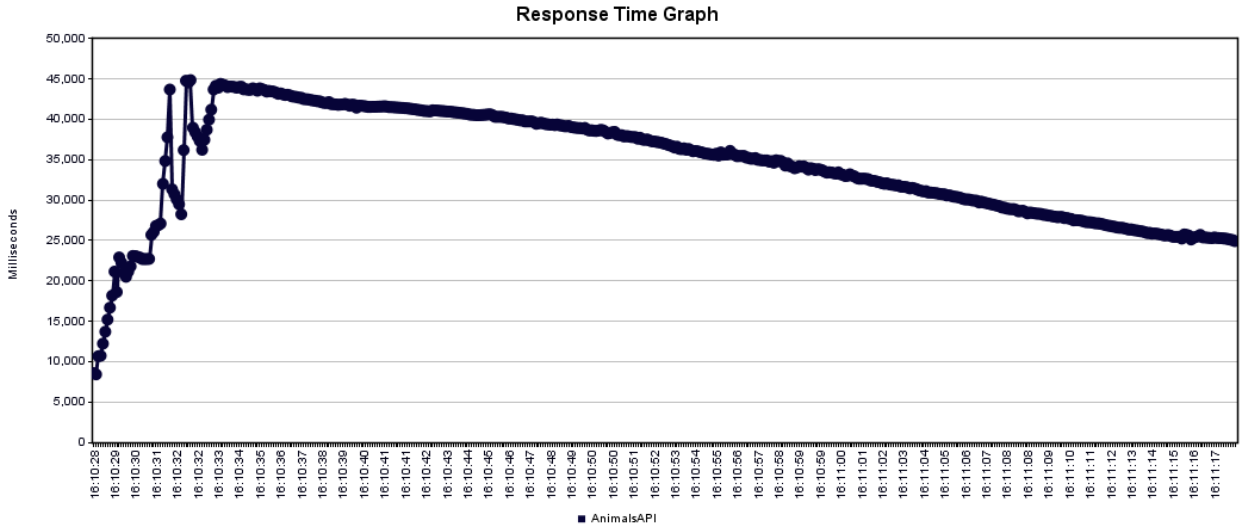


Figura 4.3: Tiempo de Respuesta de la API de Animales

4.1.4 600 solicitudes http

Al ejecutar 600 peticiones simultáneas o más se perdió gran cantidad de los paquetes enviados, esto se traduce en una sobrecarga por el lado del servidor, es decir, no posee la capacidad para poder procesar todas las solicitudes, por lo que comenzarán a perderse.

4.2 Resultados de las Predicciones

4.2.1 Tiempo de Entrenamiento

El proceso de entrenamiento toma un tiempo considerable generalmente. En el caso de los modelos aquí usados fueron 5 minutos para cada uno el cual no fue mucho debido a que no se contaba con un dataset tan amplio, lo cual se ve reflejado en los resultados, es decir que no son del todo fiables al menos para este proyecto, debido al tiempo con el que se contaba y adicionalmente debido a que cada imagen debe ser propiamente etiquetada para ser usada en el entrenamiento y para esto se debe contar con un especialista en el tema de psicología animal.

4.2.2 Proceso de Entrenamiento

En las figuras 4.4 y 4.5, se tienen dos planos con líneas rojas y verdes. El plano de la izquierda representa la precisión, la línea roja demuestra cómo esta métrica cambió

durante la etapa de prueba, mientras que la línea verde representa el cambio de la precisión durante la etapa de entrenamiento. En el plano de la derecha se representa la pérdida, la línea roja muestra el progreso de esta métrica durante la etapa de prueba y la línea verde representa la pérdida en la etapa de entrenamiento. En ambos casos, mientras el entrenamiento avanza, la precisión aumenta, al contrario que la pérdida, la cual disminuye.

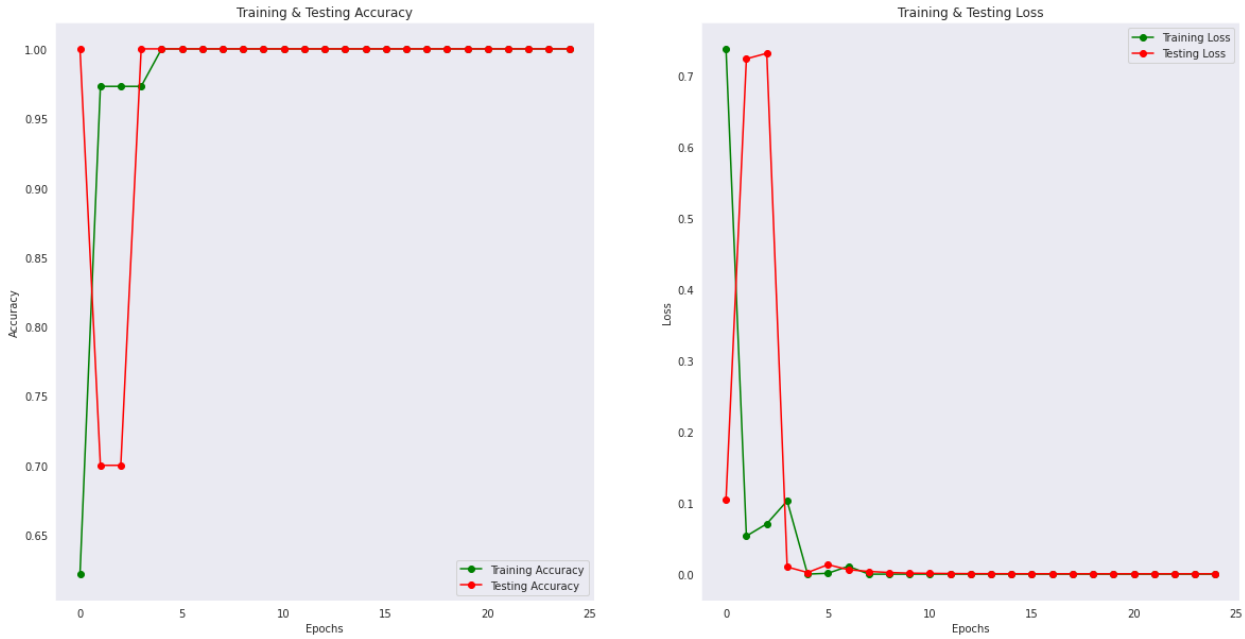


Figura 4.4: Desempeño y pérdida durante el entrenamiento y testeo del modelo de aprendizaje

En la figura 4.6 se tiene una matriz o mapa de calor. Cada celda representa la probabilidad de que al realizarse las predicciones se pueda dar una confusión entre diferentes etiquetas. En el marco de la gráfica se encuentran las etiquetas usadas para el modelo, la diagonal representará los aciertos del modelo para cada etiqueta, mientras que el resto de celdas son los errores al ejecutar la predicción, o dicho de otra manera, la probabilidad de que se confunda un elemento con otro.

Las figuras 4.7 y 4.8 contienen imágenes con leyendas en la parte superior en donde *Predicted Tree* muestra el resultado predecido por el algoritmo, y *Actual Tree* es el resultado que se debería obtener. Esto se obtiene debido a que durante el entrenamiento el algoritmo reserva un grupo de imágenes que no se usa en esta fase, más bien, se decide usarla en la etapa de prueba.

Las figuras **??** y 4.10 muestran el uso de la función que recibe la ruta de una imagen

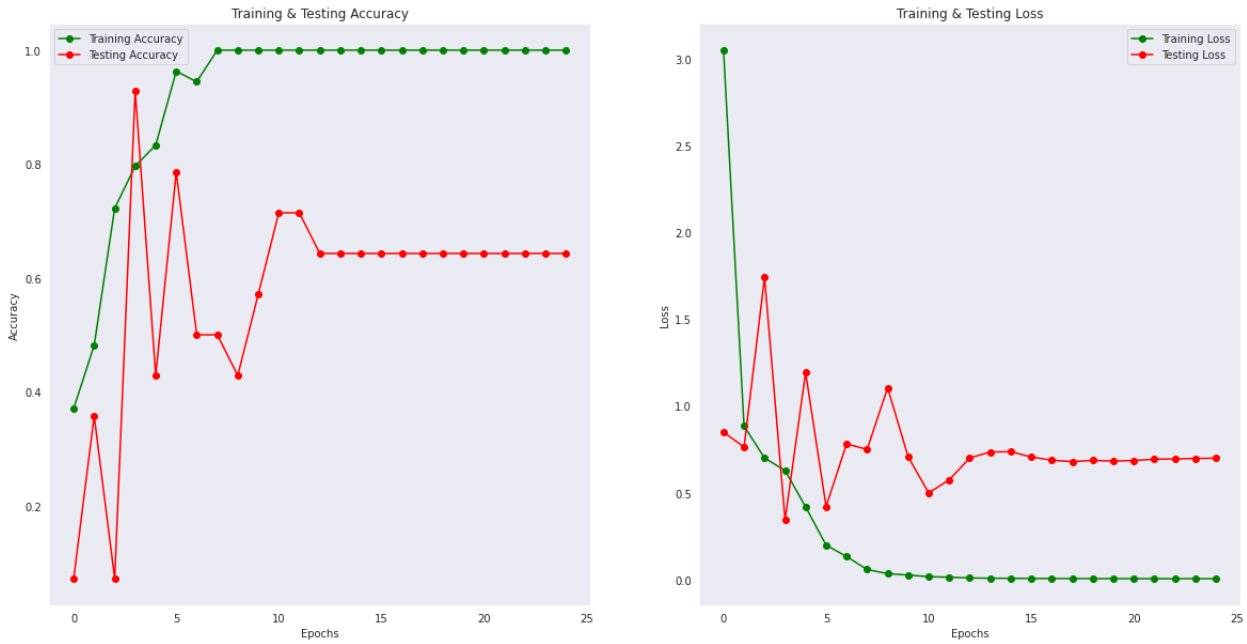


Figura 4.5: Desempeño y pérdida durante el entrenamiento y testeo del modelo de aprendizaje de gatos

para poder realizar una predicción con respecto al estado del animal. Se llama al modelo y se le envía la imagen para poder realizar la predicción.

4.3 Análisis de Satisfacción del cliente

Para medir la satisfacción del cliente se realizó una encuesta cuyas preguntas pretenden conocer la afinidad de los usuarios hacia el sistema. Se plantearon 8 preguntas las cuales se relacionan con la utilidad que les brindaría el sistema, qué tanto les agilizaría la carga respecto a los cuidados de los animales, y cuánto estarían dispuestos a pagar para adquirir la solución. Podemos ver a continuación el análisis de las respuestas brindadas.

Sobre la cantidad de personal con la que se cuenta en los establecimientos, el 60% mencionó tener menos de 5 personas que forman parte de los miembros, una cantidad alarmantemente baja, considerando la cantidad de animales a los que se debe atender y los cuidados que estos requieren. El 20% mencionó tener entre 5 y 10 miembros lo cual sigue siendo muy bajo. Un 10% respondió que cuentan con más de 10 pero menos de 20 personas mientras que el 10% restante respondió tener más de 20 pero menos de 30 miembros. Haciendo un análisis más profundo, es probable que los que respondieron contar con un máximo de 5 o 10 miembros sean parte de clínicas veterinarias, pues no

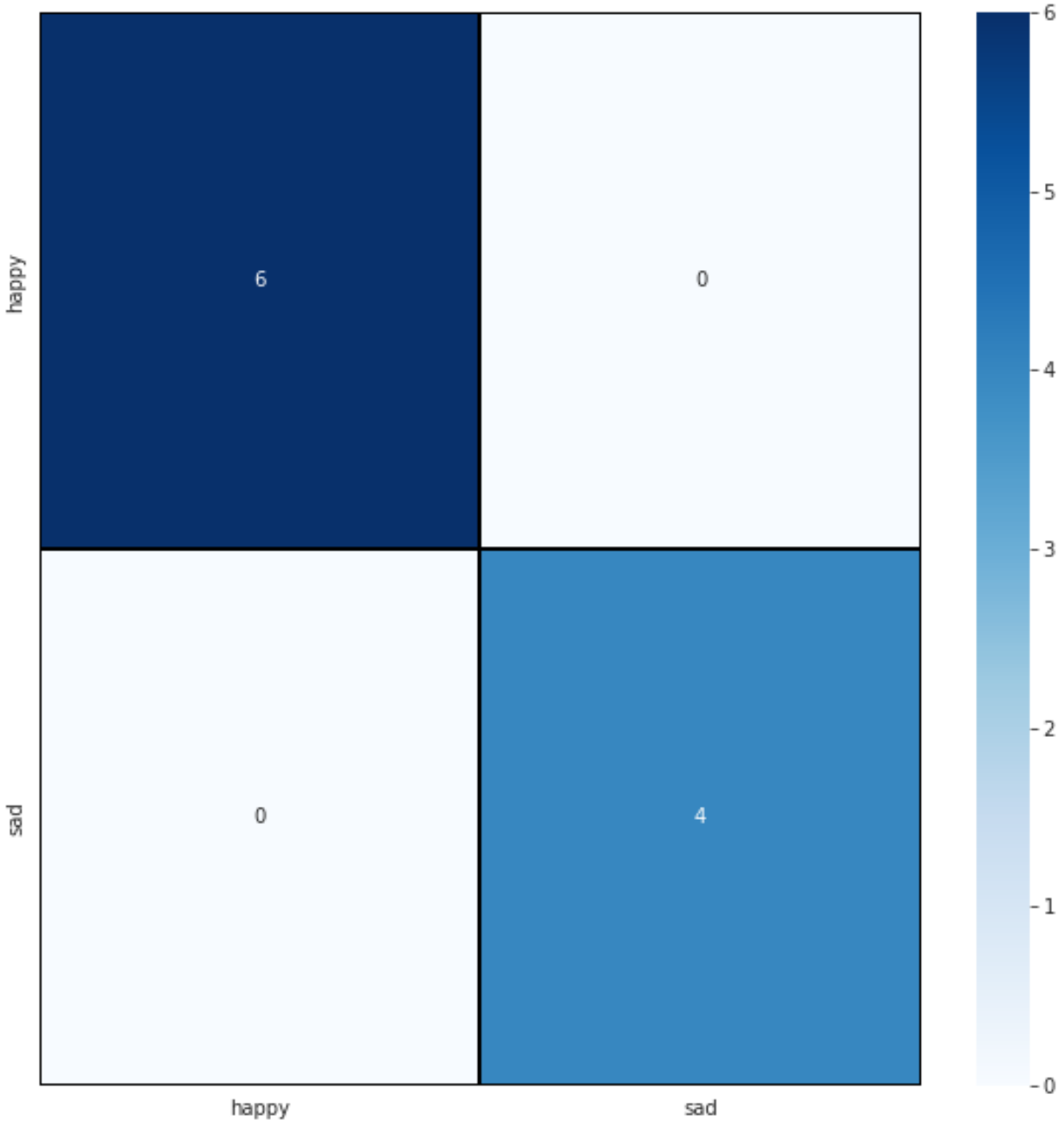


Figura 4.6: Matriz de probabilidades de predicciones del modelo de reconocimiento de emociones

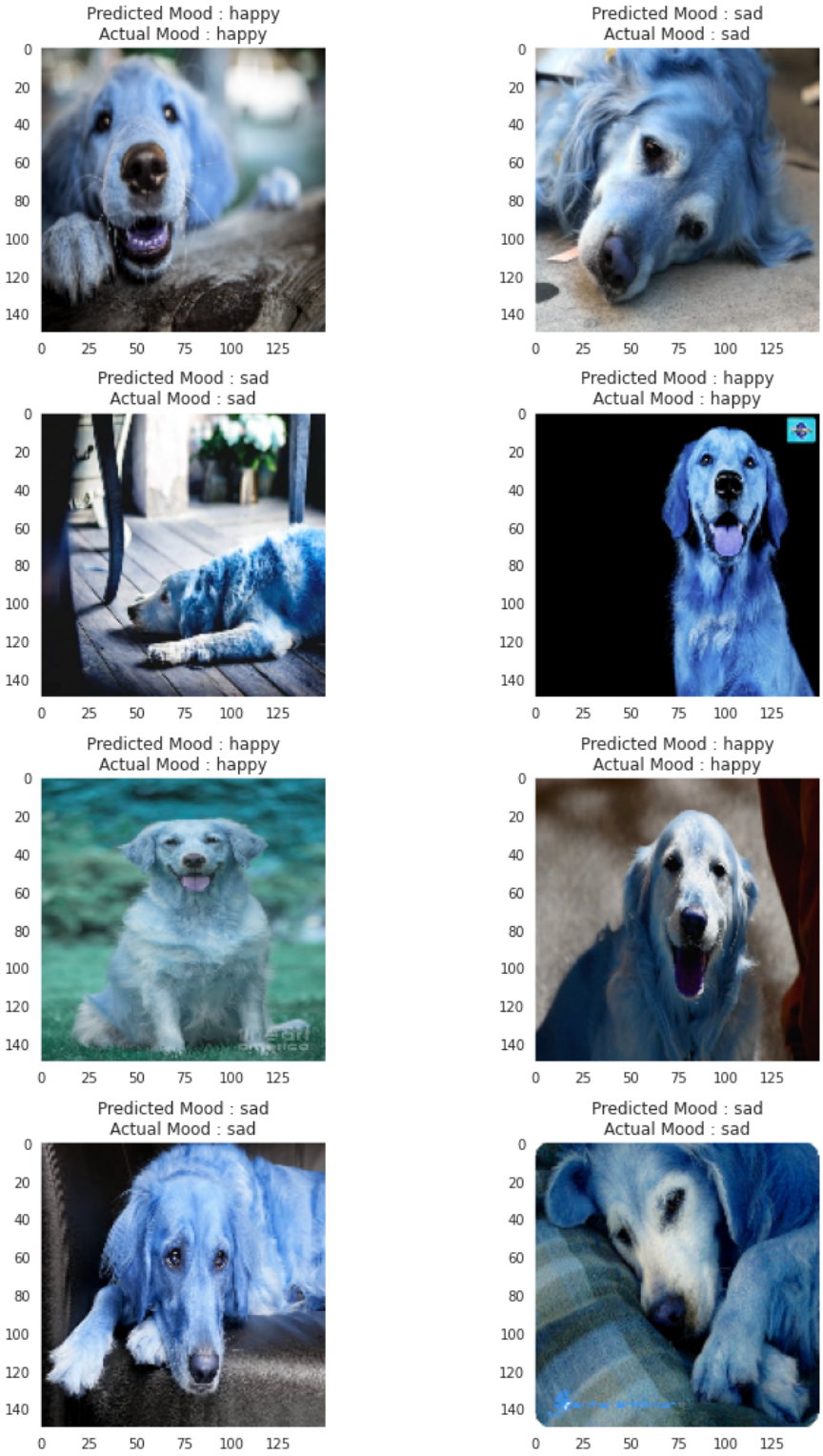


Figura 4.7: Resultados de testeo del entrenamiento del modelo de reconocimiento de emociones en perros

Figura 4.8: Resultados de testeo del entrenamiento del modelo de reconocimiento de emociones en gatos

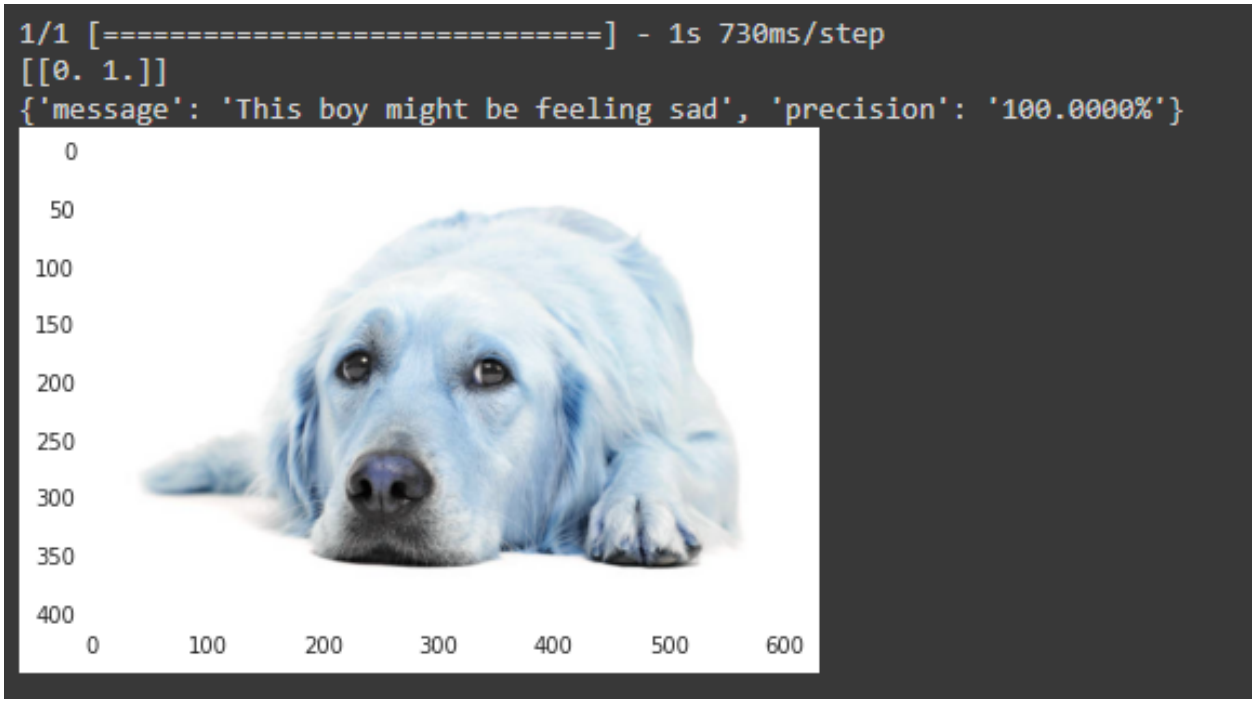


Figura 4.9: Ejecución de función para probar el modelo de perros

Figura 4.10: Ejecución de función para probar el modelo de gatos

se requiere de una gran cantidad de personal para atender estos lugares, considerando que rara vez se tiene que realizar intervenciones o internar a los animales. Mientras que aquellos que dicen contar con más de 10 miembros podrían ser refugios.

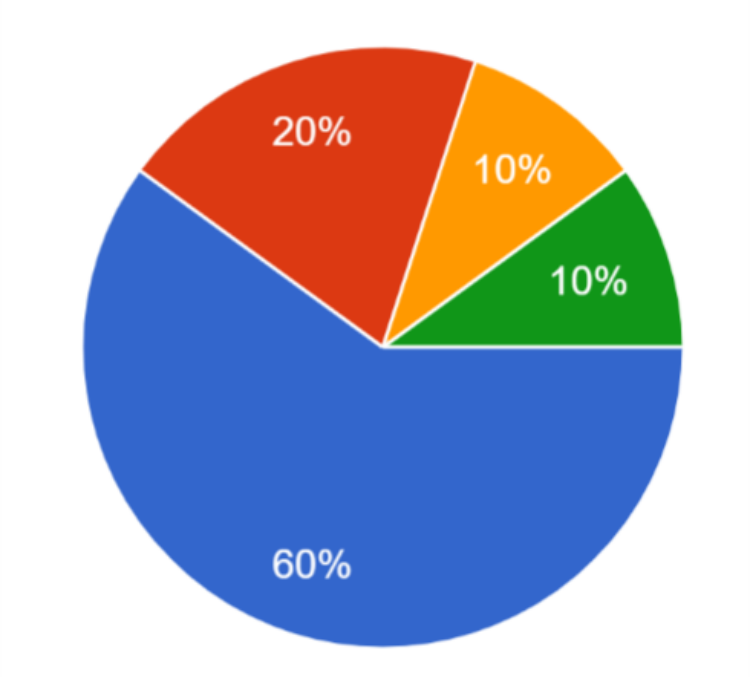


Figura 4.11: ¿Con cuantas personas, entre cuidadores, encargados y administradores, cuenta el refugio /clínica veterinaria?

Respecto a la conexión a internet, el 100% de los encuestados cuenta con conexión estable, lo cual supone una ventaja al poder realizar la comunicación entre los dispositivos hacia la nube para el intercambio de datos.

El 50% de los encuestados reportó tener conocimiento respecto a los asistentes inteligentes como Google, Alexa o Siri. No se deberá invertir demasiado tiempo en una capacitación sobre el uso de estos asistentes debido a que son muy intuitivos.

El 80% respondieron que consideran complicado estar pendiente del bienestar de todos los animales a cada momento, teniendo en cuenta la cantidad de personal que les acompaña. Se conoce sobre la situación actual de los refugios en el país, y debido a que son organizaciones sin fines de lucro y *ONGs*(Organizaciones no gubernamentales) no cuentan con los recursos necesarios para poder realizar la contrataciones adicionales ya que en su mayoría, los refugios se sostienen por las donaciones. Respecto a las veterinarias, muchas son usadas para internar a los animales luego de realizarles alguna intervención quirúrgica para esto se necesita personal que les dé los cuidados durante

el proceso de rehabilitación, sin embargo, a diferencia de los refugios, las clínicas veterinarias sí reciben ingresos económicos debido a que son negocios, de esta manera sí poseen los medios para contratar ayudantes.

El 70% respondió estar de acuerdo sobre hacer uso de una herramienta tecnológica que les facilite asegurar el bienestar de los animales en su refugio. Cabe destacar que debido a la escasez de personal muchos consideran necesario el tener otro medio para poder realizar la tarea de brindar buenos cuidados a los animales por esto la propuesta de usar una herramienta adicionales les parece llamativa. El 30% restante considera que tal vez haría uso de la herramienta, concluyendo así que sí estarían dispuestos a escuchar más sobre la propuesta y dando la posibilidad de que podrían ser convencidos.

Sobre la utilidad del sistema propuesto, el 33.3% de encuestados considera que les sería útil para que ejecute las tareas de cuidado de los animales en lugar de los encargados para que ellos puedan dedicarse a otras tareas mientras que el 66.7% restante considera que tal vez les sería útil dicho sistema. Esto se puede deber a que muchos de ellos no conocen completamente acerca de las bondades de la tecnología al relacionarse con el aprendizaje automático. Sin embargo, el hecho de que haya una posibilidad de que la vayan a usar, deja la puerta abierta para que se les pueda convencer al exponerles las ventajas que traería consigo esta propuesta.

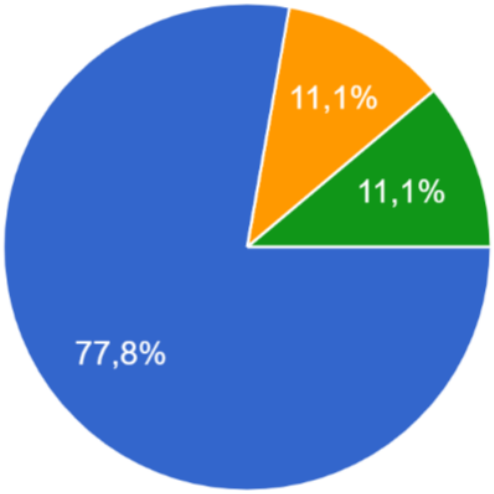


Figura 4.12: ¿Hasta cuánto estaría dispuesto a pagar por adquirir este tipo de solución?

En la figura 4.12 se puede ver cuánto estarían dispuestos a pagar por el sistema, teniendo un 77.8% que respondieron hasta $20 mensuales, un 11.1 % que respondieron hasta $40 por mes y por último un 11.1% que respondieron hasta $50 mensuales. Todos consideran importante el uso de la solución que se propone, sin embargo, al no disponer

con suficientes recursos es entendible que no estén dispuestos a pagar un precio tan alto. La propuesta no posee un precio tan elevado, debido a que se inclinó hacia el uso de software libre y tecnologías que representan costos bajos o nulos pensando en la economía del público objetivo.

CAPÍTULO 5

5. CONCLUSIONES Y RECOMENDACIONES

En este capítulo se contemplan las conclusiones y recomendaciones obtenidas a partir del desarrollo del proyecto para poder cumplir con los objetivos planteados al inicio del trabajo. A lo largo del capítulo se explica de forma sintetizada qué fue lo que limitó ciertos resultados así como cuáles son los puntos importantes del sistema propuesto.

5.1 Conclusiones

La variedad de imágenes y el etiquetado de las mismas jugaron un papel importante al momento de entrenar al sistema pues esto ayuda a garantizar que los resultados generados por el modelo sean confiables. Sin embargo, no se contaba con la cantidad de imágenes necesaria para generar resultados 100 % confiables en esta versión inicial del sistema. A pesar de esto, se puede garantizar que con más información de entrada para entrenar al modelo, se tendrá una salida fiable.

Luego de la realización de las pruebas de estrés se puede decir que el sistema demostró ser resiliente ante una cantidad de solicitudes que se vayan a ejecutar, pues, usando software en su mayoría libre, y en los casos de aquellos que no son gratis, se optó por los planes gratuitos se pudo evidenciar que hay una reacción efectiva al recibir, procesar y despachar las solicitudes por parte de los clientes. En caso de que se quiera adaptar a la aplicación para funcionar a un nivel comercial se podría realizar sin problemas.

La realización de consultas arrojó resultados sorprendentes por parte de la comunidad animalista. Las respuestas fueron positivas con respecto a una propuesta que depende 100% de la tecnología para dar cuidados a los animales. A partir de las encuestas se pudo también observar el estado en el que se encuentran muchas organizaciones que

intervienen para preservar el bienestar de las mascotas, es por esto que se alienta a ser parte del voluntariado que ejecuta labores de protección animal, donar a causas o refugios que luchan por los derechos de los animales cuando se tenga la posibilidad de donar, o simplemente ayudar a hacer consciencia transmitiendo mensajes sobre la importancia de la preservación de la fauna tanto urbana como silvestre.

El uso de *Django* como marco de referencia significó una gran ayuda al momento de implementar la aplicación web pues agilizó la tarea de construcción de las API's y la redujo a ejecutar llamados de funciones en gran parte para devolver una respuesta obtenida de la consulta a la base de datos. Además, al dar facilidad para trabajar con *Postgresql* la inserción y obtención de información resultó ser tarea fácil.

Se optó por usar 2 modelos de aprendizaje, uno por especie, de esta manera las cámaras se configurarían para enviar imágenes a rutas establecidas y se evitaría la tarea de detectar cuál es la especie que se está monitoreando. Esto resulta en menos uso de recursos computacionales, económicos y de almacenamiento.

Es cierto que existe poca investigación respecto a la salud animal, así como también es cierto que la mayoría de la gente no les suele dar la importancia que requieren en cuanto a sus cuidados. Con la propuesta aquí planteada no se pretende reemplazar el factor humano que interviene para poder cuidar a los animales, más bien se intenta facilitar la tarea de los cuidadores para que en el caso de los veterinarios, se puede invertir mayor tiempo en el tratamiento de enfermedades que requieren muchos esfuerzos, mientras que en el caso de los miembros de refugios, se pueda organizar mejor el tiempo considerando la escasez de personal con el que se cuenta.

Se espera que este estudio sea un aporte útil para la comunidad científica encargada de usar la tecnología en bien de los animales, considerando la poca investigación que existe en el campo. De esta manera se pretende que se inviertan más esfuerzos a favor de la salud de los animales. Como se menciona en la descripción del problema del proyecto, los animales suelen atacar a las personas, sin embargo, ellos no poseen una percepción racional sobre el bien o el mal de la misma manera en que lo hacen los humanos, es por esto que es necesario entender lo que sienten por medio de su lenguaje físico, y mediante este proyecto se espera lograr dicha labor.

5.2 Recomendaciones

- Se recomienda una conexión estable y con banda ancha para poder realizar la transmisión. Durante las pruebas se dieron ciertas pérdidas de paquetes en el proceso de conexión de la cámara hacia la plataforma lo cual generaba que el video se corte o se mantenga estático en ciertas secciones.

- En caso de que se vaya a añadir estados es importante tener en cuenta que se deben ejecutar cambios en el código para que el modelo reaccione de manera adecuada a la adición de etiquetas. De no realizarse las modificaciones en el algoritmo podría generar errores al correr el código, incluso después de realizar el entrenamiento. Esto derivará en errores en los resultados, es decir, no serán confiables.

- La recopilación de información y el etiquetado de las imágenes representan un rol crucial al momento de ejecutar las predicciones, es por esto que el proceso de etiquetar se debe realizar con detenimiento y junto con un experto en el tema. Es necesario que cada imagen sea analizada detenidamente para poder clasificarse y usada en el entrenamiento.

- Es importante tener en cuenta la escalabilidad del sistema, es decir, su capacidad de adaptarse a un aumento en la cantidad de animales a evaluar. Para ello, se recomienda considerar la opción de agregar servidores adicionales o utilizar la nube para almacenar la información generada por la plataforma, con el fin de evitar posibles cuellos de botella.

- Se recomienda explorar la integración con otras herramientas o aplicaciones, especialmente aquellas destinadas a la gestión y el control de refugios para animales. De esta manera, se podría obtener una mayor eficiencia en la toma de decisiones, así como una mejor coordinación y seguimiento de los animales en cuestión.

5.3 Líneas Futuras

La propuesta desarrollada en esta tesis es un avance significativo en el campo de la tecnología de reconocimiento de imágenes y la inteligencia artificial. Sin embargo, es importante destacar que aún existen áreas que pueden ser mejoradas en futuras investigaciones. A continuación, se presentan algunos aspectos que se podrían mejorar o indagar para futuros trabajos en este campo:

- Mejorar la precisión de la predicción de estados de ánimo. La plataforma no ha demostrado un nivel de precisión tan alto en la predicción de estados de ánimo, por lo que es importante seguir investigando y mejorando los modelos de Aprendizaje Automático para aumentar aún más la precisión. Esto puede incluir la exploración de nuevas técnicas de aprendizaje automático, como el *Deep Learning*[1], con el fin de mejorar la capacidad de la plataforma para reconocer patrones aún más complejos en las imágenes y realizar predicciones de más de 3 estados de ánimo.

- Ampliar el conjunto de datos de entrenamiento. El conjunto de datos utilizado para entrenar el modelo de machine learning en esta tesis es limitado en términos de cantidad y diversidad de imágenes. Por lo tanto, es importante ampliar el conjunto de datos para mejorar la generalización del modelo a diferentes razas y tamaños de perros y gatos. Esto puede incluir la creación de una base de datos más grande y diversa de imágenes de perros y gatos, así como la recopilación de información adicional, como la edad, el tamaño y la raza de los animales, que puede ser utilizada para mejorar la precisión de la predicción de estados de ánimo.

- Integrar otras fuentes de información. Además de imágenes, otros tipos de información, como audio o video, pueden ser utilizados para predecir los estados de ánimo de perros y gatos. Por lo tanto, es importante investigar la integración de otras fuentes de información en la plataforma. Por ejemplo, el análisis de audio puede proporcionar información adicional sobre los sonidos emitidos por los animales, como ladridos o maullidos, que pueden ser utilizados para identificar ciertos estados de ánimo. Además, el análisis de video puede brindar conocimiento sobre alguna dolencia que pueda detectarse solo en el movimiento corporal del individuo.

[1] https://www.sas.com/es_ar/insights/analytics/deep-learning.html

- Evaluar la eficacia de la plataforma en diferentes entornos no solo uno controlado. La plataforma desarrollada en esta tesis se ha probado en un entorno controlado. Sin embargo, es importante investigar su aplicación en entornos reales y en diferentes culturas para evaluar su eficiencia y qué tan bien funciona con varios individuos o animales que no sean precisamente mascotas educadas que no presenten comportamientos inesperados.

BIBLIOGRAFÍA

[1] S. Coren and S. Hodgson, *Entiende a tu perro para Dummies*. Para Dummies, Para Dummies, 2010.

[2] J. Fatjó, “Estudio “el nunca lo haría” de la fundación affinity sobre el abandono, la pérdida y la adopción de animales de compañía en españa 2018,” 2018.

[3] F. J. C. González, “La declaración universal de los derechos del animal,” in *Derecho Animal. Forum of Animal Law Studies*, vol. 9, pp. 143–146, 2018.

[4] M. J. Chible Villadangos, “Introducción al derecho animal: Elementos y perspectivas en el desarrollo de una nueva área del derecho,” *Ius et Praxis*, vol. 22, no. 2, pp. 373–414, 2016.

[5] C. Johnson and T. Grandin, *El lenguaje de los animales*. RBA Libros, 2020.

[6] S. G. Sotocina, R. E. Sorge, A. Zaloum, A. H. Tuttle, L. J. Martin, J. S. Wieskopf, J. C. Mapplebeck, P. Wei, S. Zhan, S. Zhang, J. J. McDougall, O. D. King, and J. S. Mogil, “The rat grimace scale: A partially automated method for quantifying pain in the laboratory rat via facial expressions,” *Molecular Pain*, vol. 7, pp. 1744–8069–7–55, 2011. PMID: 21801409.

[7] L. A. Corujo, E. Kieson, T. Schloesser, and P. A. Gloor, “Emotion recognition in horses with convolutional neural networks,” *Future Internet*, vol. 13, no. 10, p. 250, 2021.

[8] R. Coyago Simbaña, “Homenaje al olvidado: Discriminación hacia los perros callejeros y la concientización de una problemática urbana por medio de una escultura,” B.S. thesis, Quito: UCE., 2015.

[9] J. D. P. Coral, *Estimación del número de caninos domésticos encontrados en las calles de ocho parroquias del centro de Quito, utilizando el método de captura y recaptura*. PhD thesis, UNIVERSIDAD CENTRAL DEL ECUADOR, 2017.

[10] S. Hantke, N. Cummins, and B. Schuller, “What is my dog trying to tell me? the automatic recognition of the context and perceived emotion of dog barks,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5134–5138, 2018.

[11] F. Eyben, K. R. Scherer, B. W. Schuller, J. Sundberg, E. André, C. Busso, L. Y. Devillers, J. Epps, P. Laukka, S. S. Narayanan, and K. P. Truong, “The geneva minimalistic acoustic parameter set (gemaps) for voice research and affective computing,” *IEEE Transactions on Affective Computing*, vol. 7, no. 2, pp. 190–202, 2016.

[12] V. Franzoni, A. Milani, G. Biondi, and F. Micheli, “A preliminary work on dog emotion recognition,” in *IEEE/WIC/ACM International Conference on Web Intelligence-Companion Volume*, pp. 91–96, 2019.

[13] R. O. Sinnott, U. Aickelin, Y. Jia, E. R. Sinnott, P.-Y. Sun, and R. Susanto, “Run or pat: Using deep learning to classify the species type and emotion of pets,” in *2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, pp. 1–6, 2021.

[14] D. Morris, *Catwatching*. Ebury, 2002.

[15] E. M. Carozza, “Understanding the cat,” *TODAYSVETERINARYNURSE.COM*, 2018.

[16] J. L. Fermo, M. A. Schnaider, A. H. P. Silva, and C. F. M. Molento, “Only when it feels good: Specific cat vocalizations other than meowing,” *Animals*, vol. 9, no. 11, p. 878, 2019.

[17] C. A. Litchfield, G. Quinton, H. Tindle, B. Chiera, K. H. Kikillus, and P. Roetman, “The ‘feline five’: An exploration of personality in pet cats (felis catus),” *PLoS One*, vol. 12, no. 8, p. e0183455, 2017.

[18] E. Faure and A. Kitchener, “An archaeological and historical review of the relationships between felids and people. anthrozoös, 22 (3), 221–238,” 2009.

[19] S. Schötz, *The secret language of cats: how to understand your cat for a better, happier relationship*. Harlequin, 2018.

[20] P. L. Bernstein, “The human-cat relationship,” in *The welfare of cats*, pp. 47–89, Springer, 2007.

[21] P. Bateson, “Behavioural development in the cat,” *The Domestic Cat: The Biology of its Behaviour*, pp. 201–212, 2013.

[22] J. Hulick, “How to speak cat: A guide to decoding cat language by aline alexander newman,” *Bulletin of the Center for Children’s Books*, vol. 68, no. 8, pp. 414–414, 2015.

[23] P. MILLER, “Whisker whispers,” *Association of Animal Behavior*, 2000.

[24] C. Tavernier, S. Ahmed, K. A. Houpt, and S. C. Yeon, “Feline vocal communication,” *Journal of veterinary science*, vol. 21, no. 1, 2020.

[25] S. Neethirajan, “Happy cow or thinking pig? wur wolf—facial coding platform for measuring emotions in farm animals,” *AI*, vol. 2, no. 3, pp. 342–354, 2021.

[26] X. Mu, S. Li, and P. Haipeng, “A review of face recognition technology,” *IEEE Access*, vol. PP, pp. 1–1, 07 2020.

[27] D. L. Alonso-Sierra, Juan David Castaño-Saavedra, “Sistema de reconocimiento facial para control de acceso a viviendas,” *Fundación Universitaria de Ciencias de la Salud*, 2019.

[28] D. E. Espinoza Olguín and P. I. Jorquera Guillen, “Reconocimiento facial,” *PONTIFICIA UNIVERSIDAD CATOLICA DE VALPARAÍSO FACULTAD DE INGENIERIA, VALPARAISO*, 2015.