UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

DESARROLLO DE UN MODELO DE RECONOCIMIENTO DE GANADO BOVINO USANDO DEEP LEARNING.

CABRERA LAPO BRYAN GERMAN
INGENIERO DE SISTEMAS

MACHALA
2022

UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

Desarrollo de un modelo de reconocimiento de ganado bovino usando deep learning.

CABRERA LAPO BRYAN GERMAN
INGENIERO DE SISTEMAS

MACHALA
2022

UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

TRABAJO TITULACIÓN
PROPUESTAS TECNOLÓGICAS

Desarrollo de un modelo de reconocimiento de ganado bovino usando deep learning.

CABRERA LAPO BRYAN GERMAN
INGENIERO DE SISTEMAS

RIVAS ASANZA WILMER BRAULIO

MACHALA, 25 DE FEBRERO DE 2022

MACHALA
2022

tesis bryan cabrera

INFORME DE ORIGINALIDAD

7%	7%	0%	1%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	ichi.pro Fuente de Internet	1%
2	repositorio.ug.edu.ec Fuente de Internet	1%
3	dspace.utb.edu.ec Fuente de Internet	1%
4	hdl.handle.net Fuente de Internet	1%
5	repository.unab.edu.co Fuente de Internet	1%
6	Submitted to Aliat Universidades Trabajo del estudiante	<1%
7	repositorio.utn.edu.ec Fuente de Internet	<1%
8	xdoc.mx Fuente de Internet	<1%
9	www.coursehero.com Fuente de Internet	<1%

CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

El que suscribe, CABRERA LAPO BRYAN GERMAN, en calidad de autor del siguiente trabajo escrito titulado Desarrollo de un modelo de reconocimiento de ganado bovino usando deep learning., otorga a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tiene potestad para otorgar los derechos contenidos en esta licencia.

El autor declara que el contenido que se publicará es de carácter académico y se enmarca en las dispociones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

El autor como garante de la autoría de la obra y en relación a la misma, declara que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que asume la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.

Machala, 25 de febrero de 2022

CABRERA LAPO BRYAN GERMAN
0705875482

DEDICATORIA

Dedico este trabajo a mis padres, Enrique Cabrera y Yolanda Lapo, a mis hermanos, abuelos y aquellas personas que también confiaron en mí y brindaron todo su apoyo para seguir adelante y poder culminar con éxito mi carrera profesional.

Bryan German Cabrera Lapo

AGRADECIMIENTO

Agradezco a dios por darme salud y sabiduría en cada paso que doy para cumplir mis metas, a mis padres y a mi familia por ser el apoyo y pilar fundamental durante el transcurso de mi etapa universitaria.

Además, agradezco a todo el personal docente de la Carrera de Ingeniería de Sistemas que me han brindado sus conocimientos para ser una profesional, en especial a mi tutor, Ing. Sist. Rivas Asanza Wilmer Braulio, PhD, por su tiempo y esfuerzo al guiarme durante todo el proceso de titulación.

Bryan German Cabrera Lapo

RESUMEN

Ecuador es uno de los países donde tiene un gran impacto económico en el sector ganadero, muchos de los propietarios cuentan con extensas cantidades de potreros, varios de estos han optado el uso de la tecnología para incrementar su productividad y evitar la pérdida de sus animales, debido a la poca frecuencia del control a sus animales, ya que esto demanda tiempo y mayor esfuerzo físico de sus empleados, dejando desapercibido las diferentes actividades que se deben realizar día tras día en los demás potreros. En la actualidad el uso de la inteligencia artificial sobre el área de la ganadería bovina se ha vuelto una oportunidad ventajosa, optimizando sus recursos materiales y humanos, siendo así que sus arduas ahora horas de trabajo se han convertido en minutos obteniendo así muy buenos resultados y aumentando sus ganancias. Esta investigación se enfocó en la identificación de ganado independientemente de su raza, clase, color y tamaño, mediante la utilización del aprendizaje automático y el aprendizaje profundo, que son más eficientes en el procesamiento de grandes cantidades de datos para realizar un modelo de reconocimiento de objetos, teniendo en cuenta factores como el clima, el tiempo y la resolución de la imagen. En la implementación del aprendizaje de la red neuronal, se utilizó Yolov4, que utiliza un algoritmo de detección de objetos llamado darknet, y esta arquitectura proporciona una base ajustable para adaptarla para su uso de forma personal según las necesidades de aprendizaje. Este proyecto se orientó en la creación de un modelo que permita reconocer el ganado bovino sin considerar su raza o su color, mediante una sola clase denominada “bovino”, para la recopilación de las fotografías aéreas se usó un dron modelo air 2 marca DJI, logrando un total de 1630 imágenes, con forme se ha generado cada sesión de fotografías en diferentes terrenos, estas han sido divididas en 4 conjunto de datos, permitiendo el desarrollo de 8 entrenamientos combinando y uniendo cada dataset. El hardware utilizado en el entrenamiento fueron un procesador de décima generación Intel Core i7 10750H de seis núcleos, 64 GB de RAM DDR4, una gráfica NVIDIA RTX 2060 de 6 GB GDDR6 de memoria de video junto a los sistemas operativos usados Ubuntu 24.04 LTS y Windows 10 Enterprise 21H2. Las herramientas usadas fueron las tecnologías CUDA 10.1 y cuDNN 7.6.5

para la aplicación de Yolo v4, además del lenguaje de programación Python junto con sus librerías OpenCV y Matplotlib, la herramienta de etiquetado LabelImg para la generación de las etiquetas y Jupyter Notebook como cuaderno de ejecución de sentencias de código. Se seleccionó el mejor modelo obtenido de cada entrenamiento para su evaluación en la detección, se realizaron dos tipos de pruebas, una de entrenamiento y validación, y otra de rendimiento. En la prueba de evaluación se obtuvieron los valores de precisión, sensibilidad, valor de referencia y el promedio de precisión obtenidas de los resultados durante el entrenamiento. En la prueba de rendimiento se utilizó un video tomado desde un dron desde las alturas y se realizó una evaluación manual de los resultados de detección de cada modelo. Para la obtención del modelo más optimo se utilizó una evaluación basada en una escala de Likert para aplicar una calificación de puntos de acuerdo a los resultados obtenidos en las dos pruebas aplicadas. Los resultados del modelo del entrenamiento 1 fueron satisfactorios ya que fue el que mejor calificación tuvo y mejores resultados presentó con diferentes fuentes de videos de prueba.

Palabras clave: Detección de objetos, reconocimiento de bovinos, Python, Yolo v4, NVIDIA RTX, Aprendizaje profundo, CUDA, Procesamiento de imágenes

ABSTRACT

Ecuador is one of the countries where it has a great economic impact on the livestock sector, many of the owners have large amounts of paddocks, several of these have opted to use technology to increase their productivity and avoid the loss of their animals, due to the infrequent control of their animals, since this demands time and greater physical effort from their employees, leaving unnoticed the different activities that must be carried out day after day in the other paddocks. At present, the use of artificial intelligence in the area of cattle farming has become an advantageous opportunity, optimizing its material and human resources, so that its arduous hours of work have now been converted into minutes, thus obtaining very good results and increasing your earnings. This research focused on the identification of cattle regardless of their breed, class, color and size, by using machine learning and deep learning, which are more efficient in processing large amounts of data to perform a recognition model. objects, taking into account factors such as climate, time and image resolution. In the implementation of the neural network learning, Yolov4 was used, which uses an object detection algorithm called darknet, and this architecture provides an adjustable base to adapt it for personal use according to the learning needs. This project was oriented towards the creation of a model that allows cattle to be recognized without considering their race or color, through a single class called "bovine". For the collection of aerial photographs, a DJI air 2 model drone was used. achieving a total of 1630 images, as each photo session has been generated in different terrains, these have been divided into 4 data sets, allowing the development of 8 training sessions combining and uniting each dataset. The hardware used in the training was a 10th generation Intel Core i7 10750H six-core processor, 64 GB of DDR4 RAM, an NVIDIA RTX 2060 graphics card with 6 GB GDDR6 of video memory, along with the operating systems used Ubuntu 24.04 LTS and Windows. 10Enterprise 21H2. The tools used were the CUDA 10.1 and cuDNN 7.6.5 technologies for the Yolo v4 application, in addition to the Python programming language together with its OpenCV and Matplotlib libraries, the LabelImg labeling tool for label generation and Jupyter Notebook as a notebook. execution of code statements. The best model obtained from each training was selected for its evaluation in detection,

two types of tests were carried out, one for training and validation, and another for performance. In the evaluation test, the values of precision, sensitivity, reference value and the average precision obtained from the results during training were obtained. In the performance test, a video taken from a drone from above was used and a manual evaluation of the detection results of each model was carried out. To obtain the most optimal model, an evaluation based on a Likert scale was used to apply a point rating according to the results obtained in the two tests applied. The results of the training model 1 were satisfactory since it was the one that had the best rating and presented the best results with different sources of test videos.

Keywords: Object Detection, Bovine Recognition, Python, Yolo v4, NVIDIA RTX, Deep Learning, CUDA, Image Processing

ÍNDICE DE CONTENIDO

Contenido

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La irrupción de la Inteligencia Artificial ha provocado grandes aportes en la vida del ser humano, es así que en los últimos años el uso de esta tecnología se aprovecha en varios campos con el fin de optimizar labores tradicionales y hacerlas más eficientes [1]. De esta manera su práctica se hace cada vez más evidente en distintas áreas como la medicina, asistencia virtual, entretenimiento, etc. [2]

La inteligencia artificial y el uso de tecnologías digitales se extienden al sector ganadero para mejorar el control, cuidado, y calidad de los animales, de esta manera el productor ganadero reduce costos de producción y hace más eficaz su rendimiento [3] [4] [5].

Esta investigación está enfocada en el reconocimiento del ganado bovino sin importar cual sea la raza, clase, color y tamaño, esto se realiza a través de las disciplinas de la inteligencia artificial como el aprendizaje automático y aprendizaje profundo, que son más eficaces cuando se cuenta con una gran cantidad de datos para realizar un modelo de reconocimiento de objetos, teniendo en cuenta factores como el clima, tiempo y la resolución de las imágenes.

Para la aplicación del entrenamiento de la red neuronal se utilizará Yolov4 que utiliza un algoritmo de detección de objetos llamado darknet, esta arquitectura ofrece la facilidad de poder editarla para adecuarla para usos personalizados de acuerdo a las necesidades del entrenamiento.

La obtención del conjunto de datos para el entrenamiento se lo realizará por medio de la toma de fotografías mediante un dron para generar diferentes conjuntos de datos con imágenes de alta resolución de diversas ganaderías.

Para las pruebas del modelo se utilizará un conjunto de imágenes y videos de testeo para comparar los resultados obtenidos de diversos entrenamientos del modelo.

El siguiente documento está organizado de la siguiente manera:

Capítulo 1: En este capítulo, se explican las utilidades para desarrollar un modelo de detección de bovinos, describiendo el contexto del problema, la justificación, los requisitos para el prototipo y la base de la solución propuesta.

Capítulo 2: Definiciones, se presentará en detalle la base teórica del desarrollo del modelo; también se describen los objetivos, el diseño, la implementación y las pruebas del modelo.

Capítulo 3: Exponer los resultados obtenidos de los entrenamientos del modelo, así como las conclusiones y recomendaciones presentadas en detalle según los objetivos descritos y según la solución propuesta.

1. CAPÍTULO I. DIAGNÓSTICO DE NECESIDADES Y REQUERIMIENTOS

1.1. Ámbito de Aplicación: descripción del contexto y hechos de interés

Debido a la gran cantidad de ganado bovino en la provincia de el Oro, los animales no pueden estar más de 24 horas en el mismo terreno por que el pasto es insuficiente para abastecer a numerosos bovinos dado que de esta manera, los propietarios tienden a cambiar de lugar o de potrero siendo de esta manera que la distancia entre la finca y los diferentes terrenos que cuenta cada propietario son cientos de metros de distancia, el cual se ven obligados a utilizar más recursos a consumir más el tiempo laboral de sus empleados, ya que estos tienes otras funciones en la ganadería como a la limpieza y fumigación de los terrenos, al ordeño de las vacas madres.

1.2. Establecimiento de requerimientos

Para llevar a cabo los entrenamientos con YOLOv4, se debe considerar más de un ajuste tanto en el código como en el levantamiento de una nueva data, de esta manera se convertirá en un ciclo repetitivo de acciones hasta conseguir un reconocimiento considerable, logrando alcanzar un alto porcentaje de precisión.

Se considero el siguiente orden para entrenar los modelos de reconocimiento de ganado bovino:

- Selección de Red Neuronal: en este paso se recomienda investigar o escoger cuán óptima, rápida y mayor precisión ofrezca una red neuronal al detectar objetos.
- Crear Data: al levantar una nueva data poco común, se recomienda contar con equipos acordes al proyecto previo a realizar, en este caso se utilizó un dron para generar 4 datas en diferentes ganaderías logrando capturar 1630 imágenes de alta resolución.
- Etiquetar imágenes: se utilizó una herramienta llamada LabelImg, este software nos permite cargar la carpeta donde se encuentra todas las imágenes que se utilizarán para entrenar el modelo, de manera que mostrará a cada una de las imágenes y nos permitirá colocar un cuadrado sobre cualquier objeto dentro de la misma y posteriormente asignarle una etiqueta, en este caso “bovino”.

- Entrenamiento del modelo: ajustar los parámetros del batch, de las subdivisiones, del tamaño de imágenes y canales; acordé a nuestro hardware es una de las acciones primordiales que se deben realizar ya que esto puede llegar a sobre utilizar los recursos y en otros casos podría averiar el equipo.
- Probar los resultados: al final de cada entrenamiento se procede a realizar las pruebas correspondientes con imágenes y videos, se debe probar el mismo material para todos los resultados, de esta manera tendríamos más clara la diferencia de certeza al reconocer los bovinos.

1.3. Justificación de requerimiento a satisfacer

Este proyecto se enfoca en el campo de: Tecnologías de la Información y la Comunicación y en el Eje de Investigación: Ciencia de Datos e Inteligencia Artificial establecido por la Universidad Técnica de Machala (UTMACH).

La aparición de la inteligencia artificial ha contribuido enormemente a la vida humana, razón por la cual en los últimos años esta tecnología se ha utilizado en muchos campos diferentes para mejorar la simplificación de las tareas tradicionales y hacerlas más eficientes [6]. De esta forma, su actividad se hace más visible en diversos campos como la medicina, los asistentes virtuales, el entretenimiento, etc. [7]

Dada la gran cantidad de ganado en la gobernación de El Oro, los animales no pueden permanecer más de 24 horas en la misma parcela porque los pastos no son suficientes para proporcionar muchos animales porque de esta manera los propietarios tienden a cambiar de ubicación. o rodear. De esta forma, la distancia entre la finca y los diferentes terrenos de cada propietario es de cientos de metros, lo que obliga a que se utilicen más recursos para consumir más tiempo del personal, y tienen diferentes funciones, otras pecuarias como limpieza y abrevadero. Fumigación de la tierra, y el ordeñando vacas.

Esta investigación se centra en la identificación del ganado independientemente de su raza, clase, color y tamaño, y esto se realiza a través de disciplinas de inteligencia artificial como el aprendizaje automático y el aprendizaje profundo, y es más eficaz cuando se cuentan grandes

cantidades de datos para crear un modelo de reconocimiento de objetos, teniendo en cuenta factores como el clima y el tiempo y la resolución de la imagen.

2. CAPÍTULO II. DESARROLLO DEL MODELO

2.1. Definición del prototipo tecnológico

El aprendizaje profundo es un grupo de algoritmos de aprendizaje automático que intentan diseñar modelos abstractos de alto nivel en datos utilizando estructuras computacionales que admiten transformaciones recursivas y recursivas no lineales de datos expresados como una matriz o como un tensor [8] [9].

El DL es parte de un grupo más grande de métodos de aprendizaje automático que se basan en la importación de representaciones de datos [10] [11]. La observación se puede representar de muchas formas, pero algunas representaciones facilitan la comprensión de las tareas basadas en ejemplos, y la investigación en este campo intenta identificar las mejores representaciones y cómo crear modelos para identificar estas representaciones [12] [13].

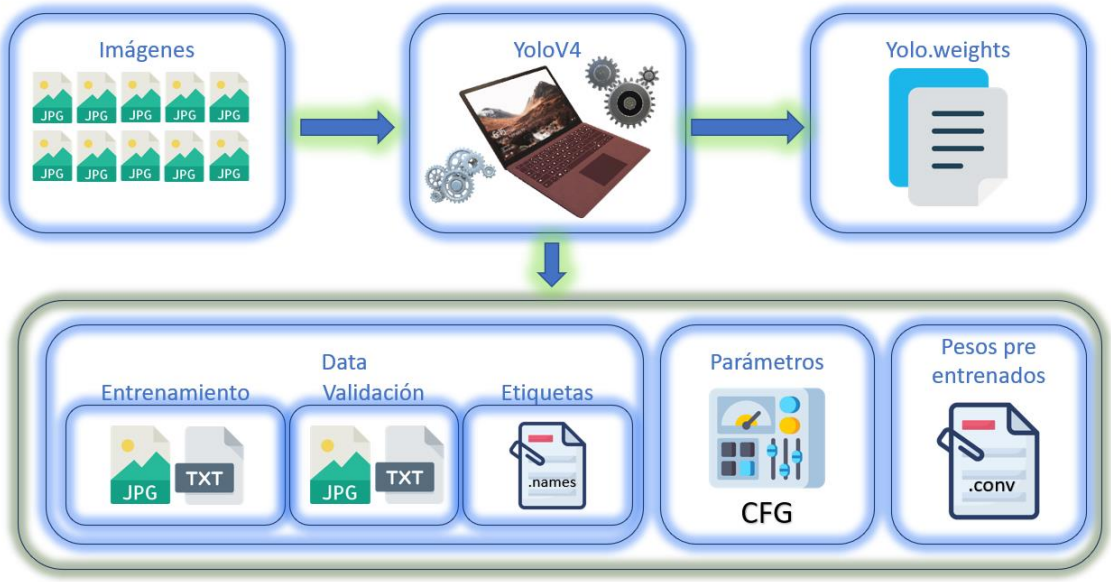
Las redes neuronales convolucionales son más eficientes que las redes neuronales tradicionales porque reducen el número de parámetros en la red y pueden procesar imágenes a una resolución más alta [14] [15].

YoloV4 está especializado en la detección de objetos y es más eficiente que otras plataformas para construir redes neuronales, ya que se entrena a partir de imágenes que contienen diferentes objetos y, a diferencia del resto, requiere una imagen de un solo objeto [16] [17]. La arquitectura del prototipo se basa en una capa de entrenamiento y otra de predicción.

La fase de entrenamiento empieza desde la creación de conjuntos de dataset de imágenes de alta resolución con el bovino, luego se realiza el etiquetado de las fotografías con la aplicación LabelImg, mediante un script de Python se separa las imágenes en tres secciones, la primera parte servirá para el entrenamiento, la segunda para la validación y la última para el testeo. Se debe editar los parámetros de configuración de la red neuronal de acuerdo a las especificaciones del hardware existente y la cantidad de imágenes que se

requiere procesar. Posteriormente se requiere descargar un archivo de pesos pre entrenados para el funcionamiento de YoloV4, del proceso de entrenamiento se obtiene un archivo con extensión “weights” con la red neuronal ya procesada y lista para ser utilizada en la detección de bovinos.

Figura 1.Arquitectura de la red neuronal



Fuente: Elaboración propia

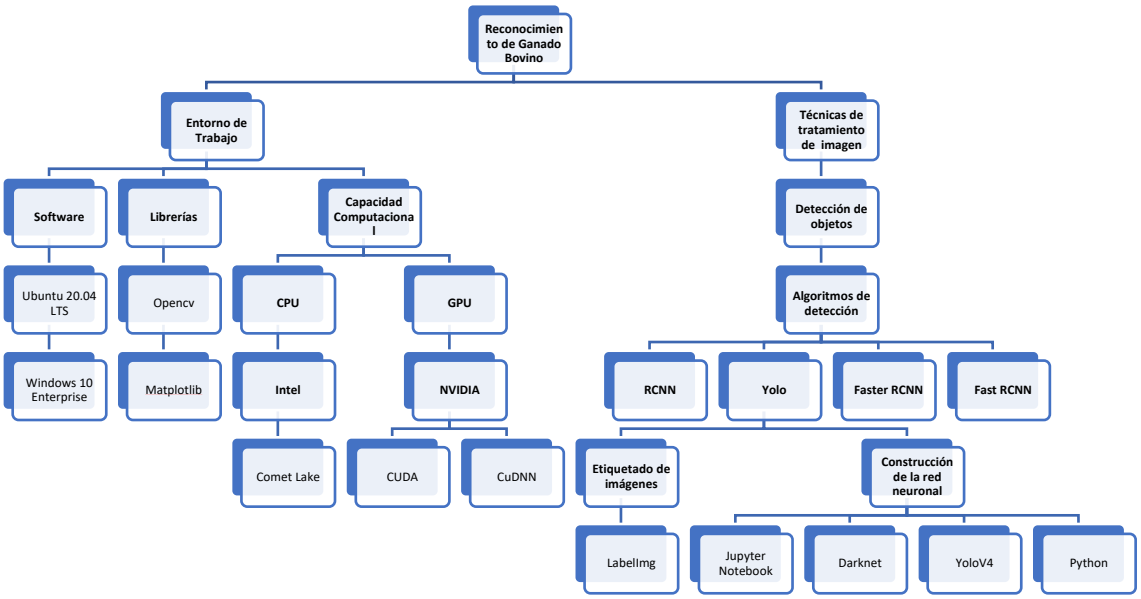
La fase de predicción abarca desde la captura de archivos de videos o imágenes que serán receptados mediante un script de Python el cual estará conectado a la red neuronal ya entrenada que hará la detección y conteo del ganado bovino que será almacenado en un archivo de salida con los resultados obtenidos.

Figura 2. Arquitectura de detección de bovinos

Fuente: Elaboración propia

2.2. Fundamentación teórica del prototipo

Figura 3. Mapa mental de la Fundamentación Teórica del Prototipo



Fuente: *Elaboración propia*

2.2.1. Entorno de Trabajo

2.2.1.1. Software

2.2.1.1.1. Ubuntu 20.04 LTS

Desarrollada por la empresa Canonical, Ubuntu es una distribución de GNU/Linux que es derivada de otra llamada Debian, este sistema operativo es de código abierto e incluye herramientas y programas que son software libre [18].

Es una de las distribuciones más utilizadas en entornos de escritorio y servidor, y Ubuntu incluye dos modelos de actualización para su sistema, una versión regular con nueve meses de soporte y otro modelo con cinco años de soporte [19].

Ubuntu es muy utilizada para estaciones de trabajo en ciencia de datos e inteligencia artificial debido a su mejor compatibilidad con los controladores de las tarjetas gráficas NVIDIA.

2.2.1.1.2. Windows 10 Enterprise

Es un sistema operativo privativo desarrollado por la multinacional Microsoft, se anunció a finales del tercer cuatrimestre del año 2014, sucediendo a Windows 8.1 y Windows 7. Esta versión incluye nuevas características como un renovado menú de inicio, la interfaz Continuum, Fluent Design, escritorios virtuales entre otros. La versión Enterprise está diseñada para los departamentos de TI de las empresas y para la administración de redes corporativas [20].

2.2.1.2. Librerías

2.2.1.2.1. OpenCV

Es una biblioteca de uso común para tareas de inteligencia artificial como machine learning, procesamiento de imágenes en tiempo real, reconocimiento y reconocimiento facial. Contiene más de 2500 algoritmos ideales para todo tipo de aplicaciones, y tiene una enorme compatibilidad en sistemas operativos como Windows, Mac OS y Linux, en aplicaciones como Android, iOS, Python, CUDA y más [21] [22].

2.2.1.2.2. Matplotlib

Es una librería completa para la creación de graficas estáticas, animadas e interactivas en el lenguaje de programación Python, dentro de sus funciones está el de crear gráficos de alta calidad, figuras interactivas con funciones de hacer zoom, desplazamiento, actualización de refresco; personalización de estilos y diseño visual, exportación a múltiples formatos, compatibilidad con Jupyter Notebook, entre otros [23].

2.2.1.3. Capacidad Computacional

2.2.1.3.1. CPU

La unidad central de procesamiento es una de las partes vitales de un computador, esta se encarga del procesamiento de datos y la ejecución de cálculos aritméticos y/o lógicos de las instrucciones de un algoritmo. La CPU

cuenta con una unidad aritmética lógica (ALU), una unidad de control, memorias Cache y DRAM [24].

2.2.1.3.1.1. Intel

Es uno de las mayores fabricantes de semiconductores a nivel mundial, esta compañía una de las creadoras e impulsoras de los procesadores de la serie x86 usados en la mayoría de computadoras de escritorio, portátiles, estaciones de trabajo, y servidores a nivel mundial [25].

2.2.1.3.1.1.1. Comet Lake

Es la arquitectura de microprocesadores Intel Core correspondiente a la décima generación de procesadores x86 de la compañía, están fabricados bajo el tercer proceso de refinamiento de Skylake a catorce nanómetros, sucede a la familia Coffe Lake y se anunció durante la mitad del tercer cuatrimestre de 2019 con la gama de procesadores móviles de bajo consumo, las series de alto rendimiento para laptops y las versiones de escritorio se anunciaron durante el segundo cuatrimestre de 2020 [26]. La serie móvil cuentan con un máximo de ocho núcleos y dieciséis hilos mientras que las versiones de escritorio cuentan con un máximo de diez núcleos y veinte hilos con una frecuencia máxima de 5.3 GHz.

2.2.1.3.2. GPU

La unidad de procesamiento gráfico es la encargada de realizar las tareas de procesado de gráficos y operaciones de cálculo de coma flotante para aliviar la carga de trabajo de la unidad central de procesamiento en programas que requieran realizar los procesos 3D y 2D como los videojuegos y aplicaciones de renderizado [27] [28].

2.2.1.3.2.1. NVIDIA

Es una de las empresas multinacionales especializadas en el desarrollo de gráficas y tecnologías de circuitos integrados para computadores de escritorio, portátiles, estaciones de trabajo y servidores [29]. Está involucrada en el desarrollo de tecnologías sobre el machine learning e inteligencia artificial.

2.2.1.3.2.1.1. CUDA

CUDA es un modelo de programación en paralelo desarrollado por NVIDIA que le permite utilizar la unidad de procesamiento gráfico a su máxima capacidad para aumentar el rendimiento de las aplicaciones desarrolladas [30] [31].

2.2.1.3.2.1.2. CuDNN

Es un conjunto de librerías de primitivas aceleradas por una unidad de procesamiento gráfico para su aplicación en redes neuronales profundas. Permite implementaciones altamente optimizadas de operaciones estándar, como convolución hacia adelante y hacia atrás, capas de agrupación, normalización y activación [32].

2.2.2. Técnicas de Tratamiento de Imágenes

2.2.2.1. Detección de Objetos

La detección de objetos es una tecnología informática relacionada con la visión por computadora y el procesamiento de imágenes que intenta detectar los estados de los objetos semánticos de una clase en particular (como personas, edificios o automóviles) en objetos de imágenes y videos digitales [33] [34]. Los campos más avanzados de detección de objetos incluyen la detección de rostros y la detección de personas. El descubrimiento de objetos tiene aplicaciones en muchas áreas de la visión artificial, incluida la recuperación de imágenes y la videovigilancia [35].

El método de detección de objetos fundamentalmente está constituido por tres partes: el diseño de características, los cuadros de detección y el diseño de clasificador. El diseño de características es la parte en donde se configura y se extraen características de la red neuronal [36]. Los cuadros de detección incluyen métodos de búsqueda exhaustiva y selectiva, así como también una Red propuesta por Regiones (RPN) que está basada en Deep Learning [37].

Para la detección de objetos se debe elaborar un conjunto de datos con imágenes que contengan la información de los objetos a identificar, luego se realiza el etiquetado de cada objeto que se vaya encontrando en cada imagen del conjunto de datos detallando a que clase pertenece y que posición se

encuentra cada objeto, una de las herramientas utilizadas en este campo es la aplicación LabelImg [38].

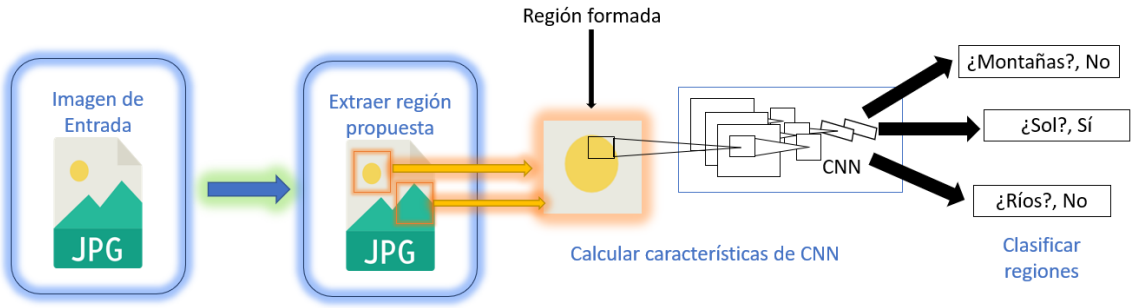
2.2.2.1.1. Algoritmos de detección

2.2.2.1.1.1. R-CNN

Este modelo de detección de objetos se basa en la búsqueda selectiva que utiliza un método de extracción de solo 2000 regiones de la imagen y de la cual se denomina propuesta de región [39]. En lugar de clasificar una enorme cantidad de regiones se trabaja solamente con 2000, las propuestas de 2000 regiones se generan a partir de un algoritmo de búsqueda selectiva cuyos pasos son:

- Generar una subsegmentación inicial, generamos muchas regiones candidatas.
- Usar un algoritmo voraz para combinar recursivamente regiones similares en otras más grandes.
- Usar las regiones generadas para producir las propuestas finales de regiones candidatas.

Figura 4. Regiones con funciones CNN



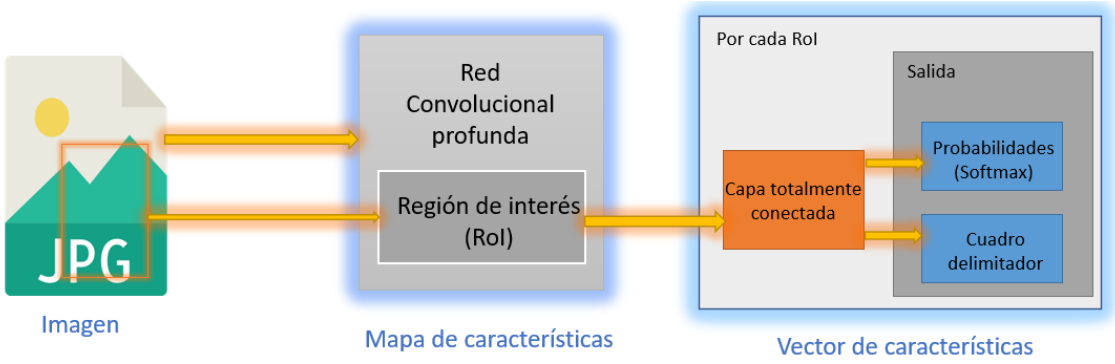
Fuente: Elaboración propia

2.2.2.1.1.2. Fast R-CNN

El enfoque de este método es similar al algoritmo R-CNN, en lugar de enviar las propuestas de la región a la CNN, envía la imagen de entrada a la CNN para la creación de un mapa de características convolucionales [39]; Desde el mapa de características convolucionales se selecciona la región de las propuestas y se altera en cuadrados mediante el uso de una capa de

agrupación RoI, se remodela en un tamaño fijo para que pueda alimentarse en una capa completamente conectada. Desde el vector de características de RoI, se usa una capa softmax para predecir la clase de la región propuesta y también los valores de desplazamiento para el cuadro delimitador. Una de las razones por la que este modelo es más rápido que R-CNN es debido a que no tiene que enviar propuestas de 2000 regiones a la red neuronal convolucional a la vez, sino que la convolución solo tiene lugar una vez por imagen y se genera un mapa de características a partir de esta.

Figura 5. Regiones con funciones Fast R-CNN



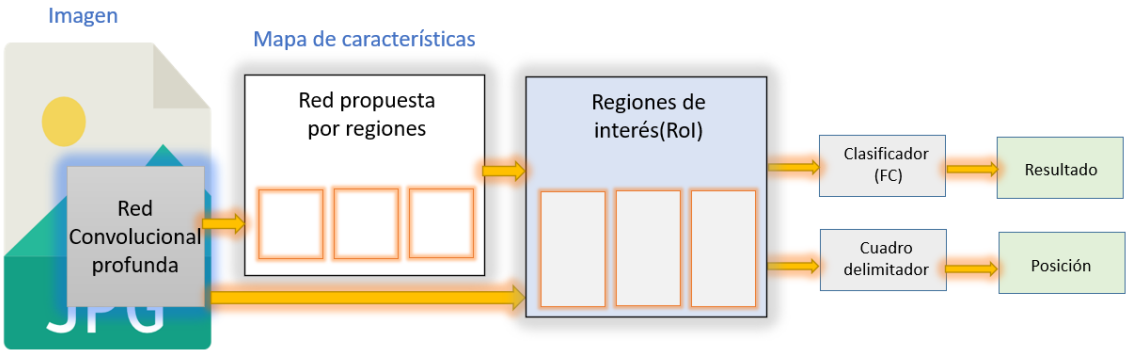
Fuente: Elaboración propia

2.2.2.1.1.3. Faster R-CNN

Los métodos anteriores usan la búsqueda selectiva para encontrar las propuestas regionales, este proceso es demasiado lento que tiene un impacto negativo en la red, por lo que este algoritmo elimina este proceso de búsqueda y permite que la red tenga un mayor aprendizaje de las propuestas de la región [39].

Es parecido al método Fast R-CNN, una imagen es proporcionada como entrada a la red convolucional que genera un mapa de características convolucionales, se usa con una red separada para la predicción de las propuestas de la región. La propuesta de región predicha se remodela mediante el uso de una capa de agrupación RoI que posteriormente se utiliza para la clasificación de la imagen dentro de la región propuesta y luego se predice los valores de compensación para los cuadros delimitadores.

Figura 6. Regiones con funciones Faster R-CNN



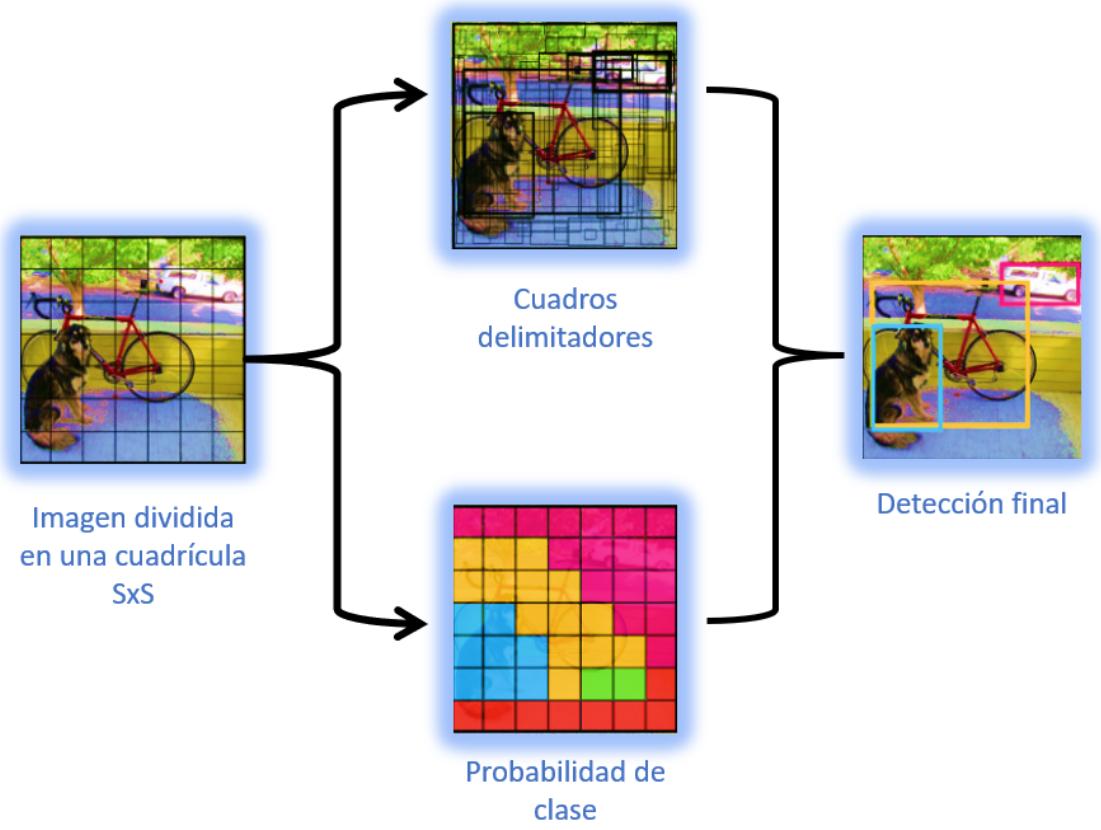
Fuente: Elaboración propia

2.2.2.1.1.4. Yolo

Este modelo utiliza una sola red convolucional para predecir los cuadros delimitadores y las probabilidades de clase para aquellos cuadros [39]. El funcionamiento de Yolo se basa en tomar una imagen y dividirla en una cuadricula SxS, en el interior de cada cuadricula se toma M cuadros delimitadores, por cada uno de los cuadros delimitadores, la red crea una probabilidad de clase y valores de desplazamiento para cada cuadro delimitador.

Los cuadros delimitadores que poseen una probabilidad de clase superior de un valor umbral son seleccionados y utilizados para ubicar el objeto dentro de la imagen.

Figura 7. Arquitectura de la red Yolo



Fuente: Elaboración propia

2.2.2.1.1.4.1. Etiquetado de Imágenes

2.2.2.1.1.4.1.1. LabelImg

Es una aplicación de escritorio escrita en Python con QT como interfaz gráfica disponibles para las plataformas Windows, MacOS y Linux que sirve para la anotación de imágenes gráficas [40].

2.2.2.1.1.4.2. Construcción de la red neuronal

2.2.2.1.1.4.2.1. Jupyter Notebook

Es un entorno de desarrollo basado en web para su utilización con cuadernos, datos y código Python, posee una interfaz intuitiva y flexible para entornos de trabajo en ciencia de datos, periodismo computacional, inteligencia artificial y computación científica [41].

2.2.2.1.1.4.2.2. Darknet

Es un marco de red neuronal open source escrito en el lenguaje de programación C con la tecnología CUDA de Nvidia, es muy rápido y sencillo de instalar además de que admite el uso tanto del procesador como de la tarjeta gráfica para hacer cálculos [42].

2.2.2.1.1.4.2.3. YoloV4

Es una bifurcación del trabajo original de Yolo v3 realizada por Alexey Bochkovskiy y lanzada al mercado en abril de 2020. Esta red está basada en Darknet y posee un valor de precisión AP del 43.5 % en el conjunto de datos COCO con una velocidad en tiempo real de 65 fps en una tarjeta gráfica Nvidia Tesla V100, en comparación con Yolo v3 el valor AP y los fps promedios tuvieron un aumento promedio de un 10 % y 12 % respectivamente [43].

2.2.2.1.1.4.2.4. Python

Es un lenguaje de programación interpretado multiplataforma que tiene como objetivo filosófico la legibilidad del código además soporta la programación imperativa, parcialmente la programación orientada a objetos y programación funcional. Python permite la creación y ejecución de scripts mediante la utilización de intérpretes y librerías que son usados en multitud de campos [44].

2.3. Metodología

Para la elaboración del dataset de imágenes se realizó la visita de varias fincas ganaderas con el objetivo de recolectar la información requerida mediante la utilización de un dron desde las alturas.

Luego se procedió a editar los parámetros de configuración de la red Yolo v4 de acuerdo al hardware disponible y se realizó varios entrenamientos con los diferentes dataset obtenidos de las fincas ganaderas.

Posteriormente se aplicaron pruebas de ejecución y testeo a los modelos entrenados para obtener las métricas necesarias para la comparación de los modelos obtenidos.

Por último, se obtiene el modelo más óptimo de la comparación de modelos para la detección de bovinos.

Figura 8. Metodología

Fuente: Elaboración propia

2.4. Objetivos del prototipo

2.4.1. Objetivos General

Desarrollar un modelo para el reconocimiento de ganado bovino aplicando algoritmos de Deep Learning.

2.4.2. Objetivos Específicos

- Seleccionar la red neuronal que permita detectar objetos mediante Deep Learning.
- Crear una data de 1630 imágenes para entrenar el modelo.
- Utilizar LabelImg en el etiquetado de las imágenes previo al entrenamiento de los modelos.
- Entrenar un modelo usando una data de 1630 fotografías para reconocer animales bovinos.
- Realizar pruebas del reconocimiento de objetos del modelo entrenado.

2.5. Diseño del prototipo

El desarrollo del prototipo esta divido en dos partes: la primera parte consta en el diseño, configuración y entrenamiento de la red neuronal y la segunda parte en la detección del ganado bovino.

2.5.1. Requisitos

A continuación, se detallan los recursos a utilizar:

Hardware:

- Marca: MSI GP65 Leopard
- Procesador: Intel Core i7 10750H Comet Lake 6C/12T 2.6 GHz – 5 GHz
- Memoria: 64 GB RAM DDR4 2666 MHz
- Gráfica: Nvidia GeForce RTX 2060 6 GB GDDR6
- Sistema Operativo: Ubuntu 20.04.3 LTS – Windows 10 Enterprise 21H2

Software:

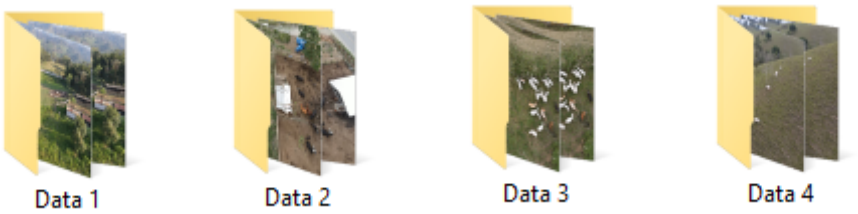
- Python
- Jupyter Notebook
- LabelImg
- OpenCV

2.5.2. Elaboración de la Dataset

Para generar la dataset, se visitó 4 fincas ganaderas que se dedican a la crianza de ganado bovino, de manera que se levantó 4 datas siendo así la

sumatoria de estas, una cantidad total de 1630 imágenes. Para obtener cada una de estas fotografías, se tuvo que hacer el uso de un dron a una altura variable de 3 hasta 45 metros de distancia desde el terreno donde se encontraban los animales, evitando así la incomodidad o susto de los mismos.

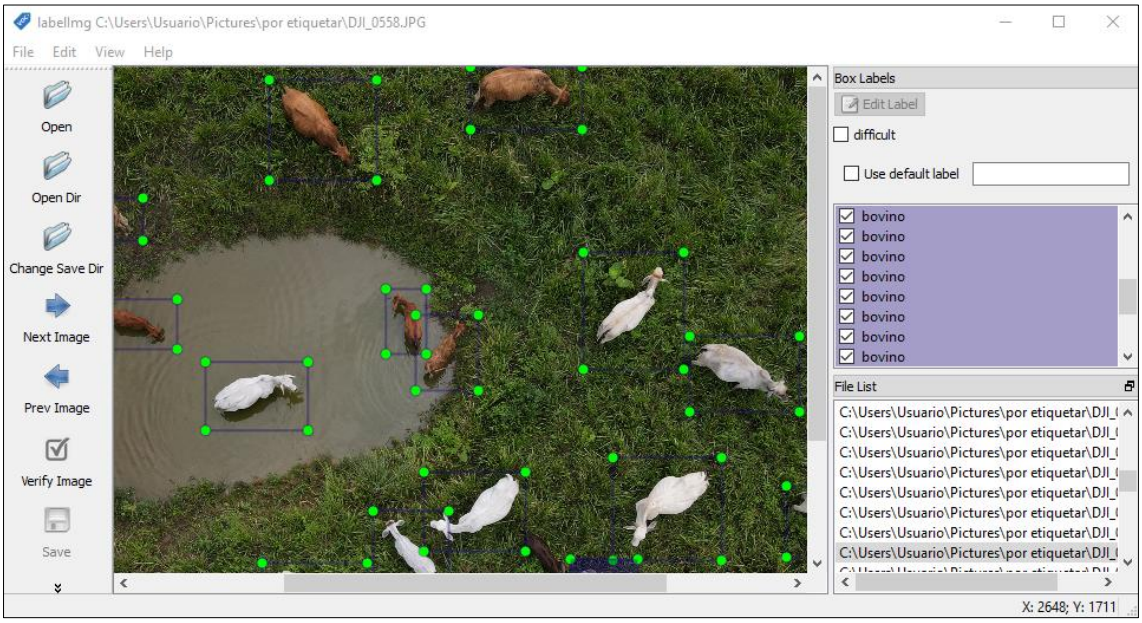
Figura 9. Elaboración de los diferentes conjuntos de datos



Fuente: Elaboración propia

Al finalizar el levantamiento de estas datas, se procedió a etiquetar a cada una de las imágenes, para aquello se utilizó la herramienta LabelImg de manera que se seleccionó las vacas madres, toros de engorde, toros reproductores y terneros de diferente raza y color, siendo así que se los identifico con la palabra “bovino”

Figura 10. Etiquetado de las imágenes con LabelImg en Windows

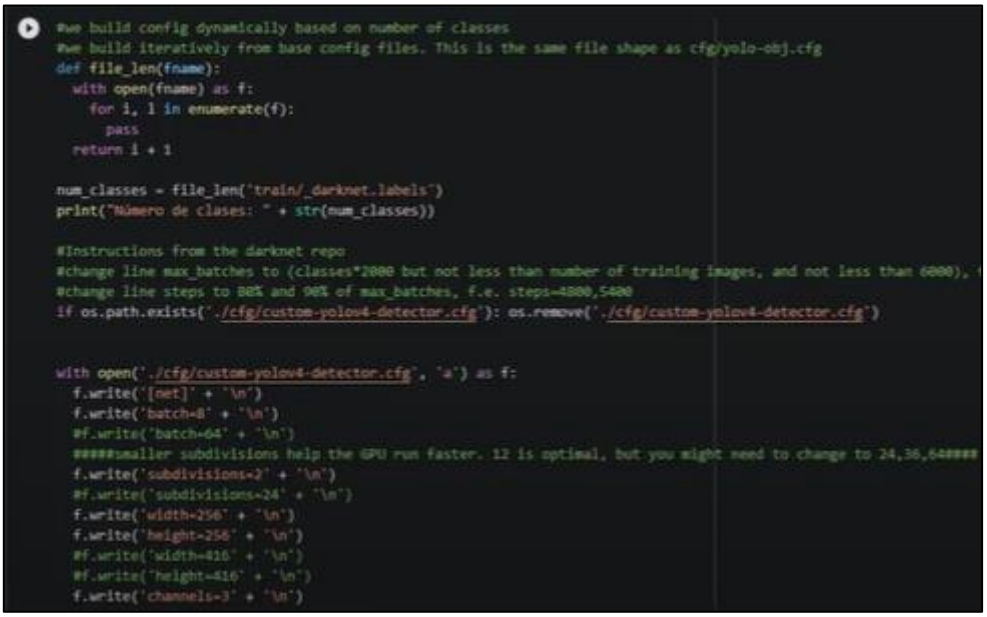


Fuente: Elaboración propia

2.5.3. Diseño y entrenamiento de la red

Al diseñar la red se consideró las capacidades del hardware, por lo tanto, se modificó los parámetros que nos ofrece el código de Yolov4 en el editor de Jupyter Notebook.

Figura 11. Configuración de los parámetros de Yolo v4



Fuente: Elaboración propia

Una vez indicada la ruta de la dataset, se modificó el número de imágenes, las subdivisiones, el tamaño y canales.

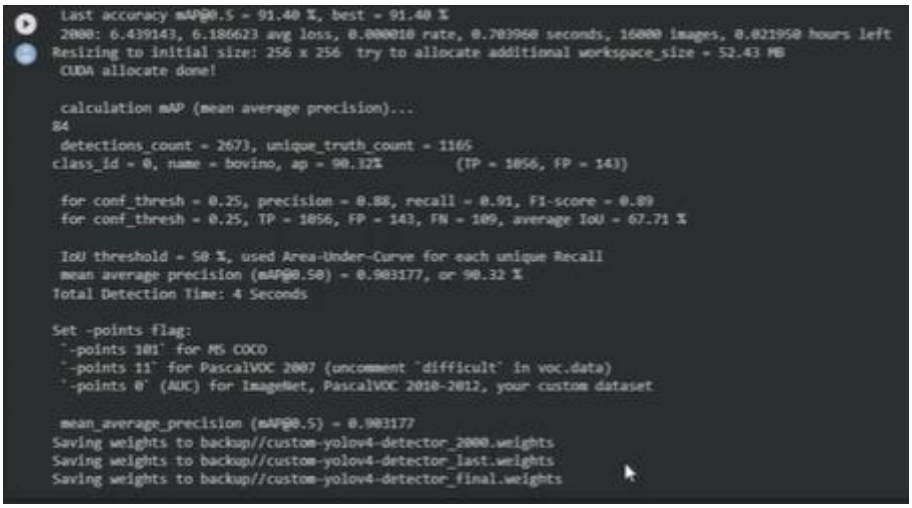
Luego de esto se procedió a entrenar el modelo, con la siguiente línea de código dentro de Jupyter Notebook.

!./darknet detector train data/abj.data cfg/custom-yolov4-detector.cfg

yolov4.conv.137 -dont_show -map

Esta línea nos indica que dentro del directorio darknet con la variable detector, se coloca el directorio train, es decir se ubican las imágenes del entrenamiento con la data obj.data, donde ya se encuentra escrita la configuración de las imágenes, cfg/custom-yolov4-detector.cfg es la configuración que se realiza antes de entrenar el modelo, y con -dont_show -map nos permite visualizar cómo va el entrenamiento.

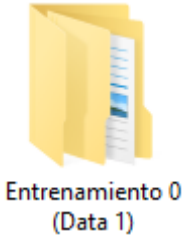
Figura 12. Finalización del entrenamiento con Yolo v4



Fuente: Elaboración propia

Siendo así que se utilizó la data 1 para realizar el entrenamiento 0, de manera que se ejecutó el código con los siguientes parámetros:

Figura 13. Carpeta donde se almaceno los archivos del entrenamiento 0



Fuente: Elaboración propia

Tabla 1. Datos del entrenamiento 0

Número de Imágenes	280
Batch	8
Subdivisiones	2
Tamaño de imágenes	256 * 256
Canales	3

Fuente: Elaboración propia

Figura 14. Estadísticas del entrenamiento 0

```
IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.436017, or 43.60 %
Total Detection Time: 3 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.5) = 0.436017
Saving weights to backup//custom-yolov4-detector_2000.weights
Saving weights to backup//custom-yolov4-detector_last.weights
Saving weights to backup//custom-yolov4-detector_final.weights
```

Fuente: Elaboración propia

Se observa que el entrenamiento 0 cuenta con una precisión del 43.60%, el cual el reconocimiento del ganado bovino es muy malo.

Figura 15. Resultados preliminares del entrenamiento 0

Fuente: Elaboración propia

Luego se realiza el entrenamiento 1 con imágenes de la data 2, a continuación, se muestran los parámetros:

Tabla 2. Datos del entrenamiento 1

Número de Imágenes	425
Batch	8
Subdivisiones	2
Tamaño de imágenes	256 * 256
Canales	3

Fuente: Elaboración propia

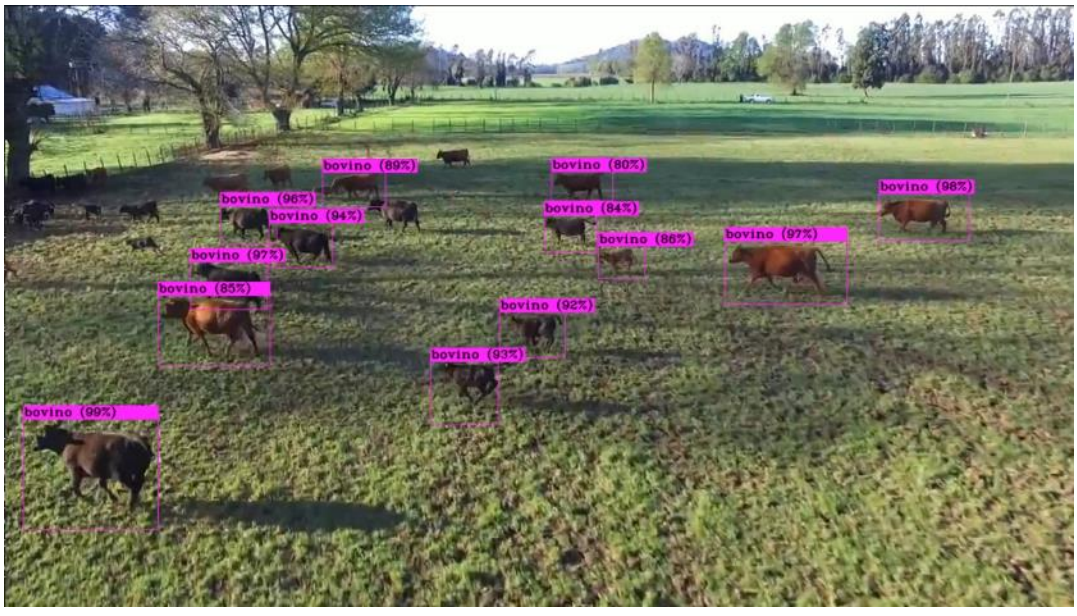
Figura 16. Estadísticas del entrenamiento 1

```
 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.929543, or 92.95 %
Total Detection Time: 2 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.5) = 0.929543
Saving weights to backup//custom-yolov4-detector_2000.weights
Saving weights to backup//custom-yolov4-detector_last.weights
Saving weights to backup//custom-yolov4-detector_final.weights
```

Fuente: Elaboración propia

Se observa que el entrenamiento 1 cuenta con una precisión del 92.95%, el cual el reconocimiento del ganado bovino es muy bueno, pero aún quedan bovinos sin ser considerados por el modelo.

Figura 17. Resultados preliminares del entrenamiento 1



Fuente: Elaboración propia

Luego se realizó el entrenamiento 2 con imágenes unificadas de la data 1 y 2, a continuación, se muestran los parámetros:

Tabla 3. Datos del entrenamiento 2

Número de Imágenes	705
Batch	8
Subdivisiones	2
Tamaño de imágenes	256 * 256
Canales	3

Fuente: Elaboración propia

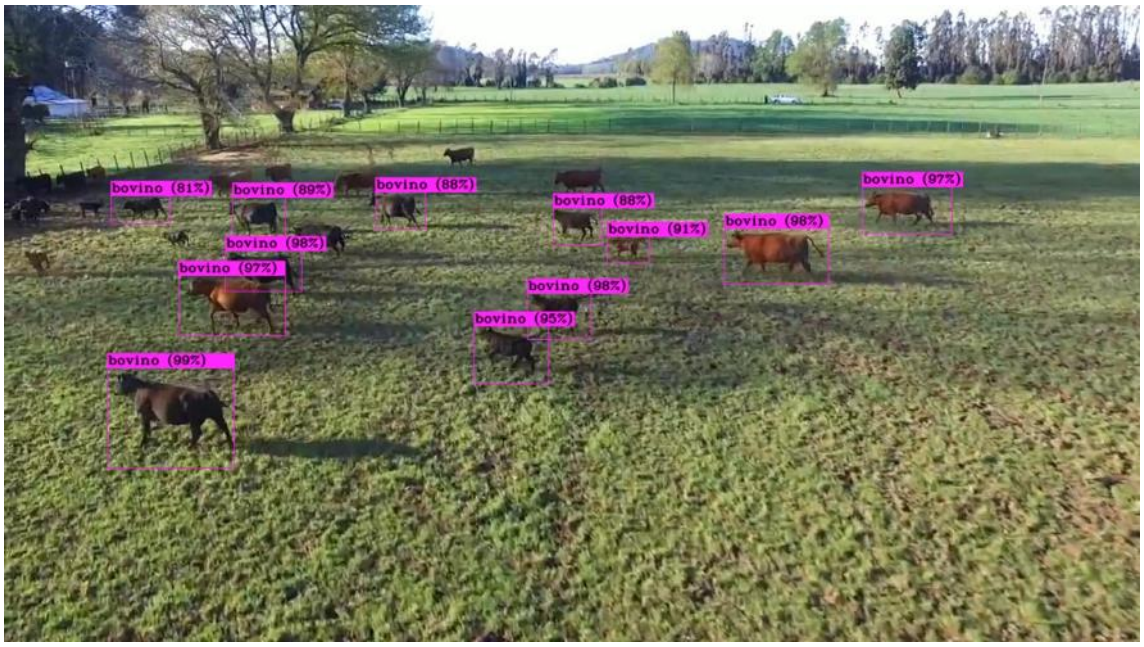
Figura 18. Estadísticas del entrenamiento 2

```
 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.590728, or 59.07 %
Total Detection Time: 3 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.5) = 0.590728
Saving weights to backup//custom-yolov4-detector_2000.weights
Saving weights to backup//custom-yolov4-detector_last.weights
Saving weights to backup//custom-yolov4-detector_final.weights
```

Fuente: Elaboración propia

El entrenamiento 2 cuenta con una precisión del 59.07%, el cual el reconocimiento del ganado bovino es bueno.

Figura 19. Resultados preliminares del entrenamiento 2



Fuente: Elaboración propia

Luego se realizó el entrenamiento 3 con imágenes de la data 3, a continuación, se muestran los parámetros:

Tabla 4. Datos del entrenamiento 3

Número de Imágenes	418
Batch	8
Subdivisiones	2
Tamaño de imágenes	256 * 256
Canales	3

Fuente: Elaboración propia

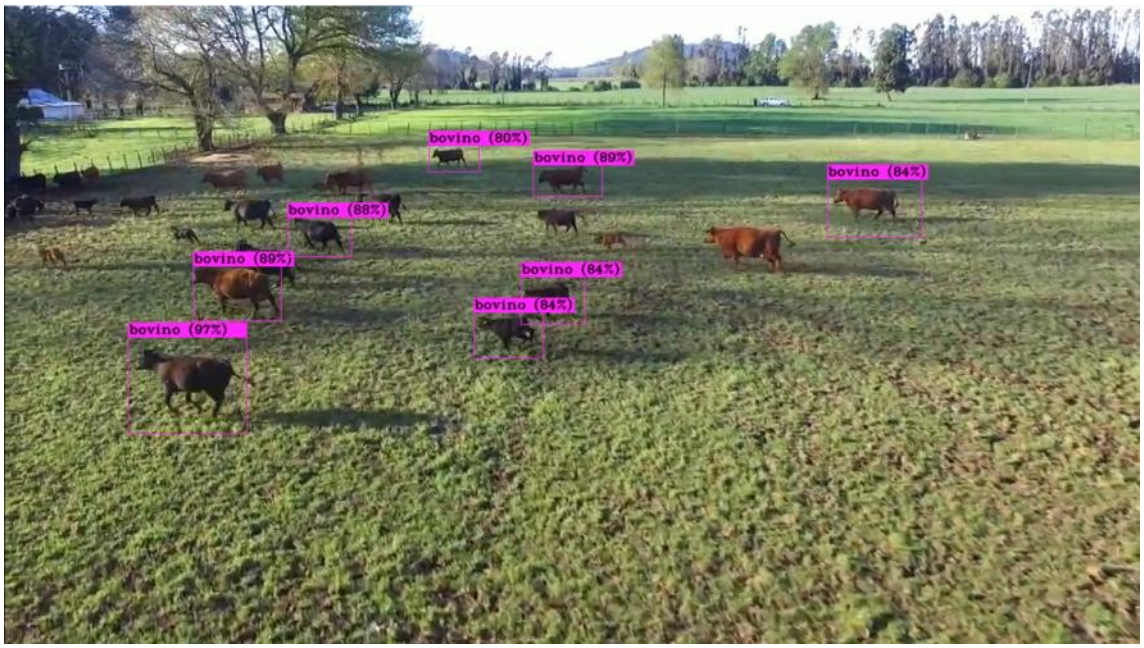
Figura 20. Estadísticas del entrenamiento 3

```
IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.895184, or 89.52 %
Total Detection Time: 2 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.5) = 0.895184
Saving weights to backup//custom-yolov4-detector_2000.weights
Saving weights to backup//custom-yolov4-detector_last.weights
Saving weights to backup//custom-yolov4-detector_final.weights
```

Fuente: Elaboración propia

El entrenamiento 3 cuenta con una precisión del 89.52%, el cual el reconocimiento del ganado bovino es regular.

Figura 21. Resultados preliminares del entrenamiento 3



Fuente: Elaboración propia

Luego se realizó el entrenamiento 4 con imágenes unificadas de la data 2 y 3, a continuación, se muestran los parámetros:

Tabla 5. Datos del entrenamiento 4

Número de Imágenes	843
Batch	8
Subdivisiones	2
Tamaño de imágenes	256 * 256
Canales	3

Fuente: Elaboración propia

Figura 22. Estadísticas del entrenamiento 4

```
IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.903177, or 90.32 %
Total Detection Time: 4 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.5) = 0.903177
Saving weights to backup//custom-yolov4-detector_2000.weights
Saving weights to backup//custom-yolov4-detector_last.weights
Saving weights to backup//custom-yolov4-detector_final.weights
```

Fuente: Elaboración propia

El entrenamiento 4 cuenta con una precisión del 90.32%, el cual el reconocimiento del ganado bovino es similar al del entrenamiento 2, pero aún existen bovinos sin ser reconocidos.

Figura 23. Resultados preliminares del entrenamiento 4

Fuente: Elaboración propia

Luego se realizó el entrenamiento 5 con imágenes de la data 4, a continuación, se muestran los parámetros:

Tabla 6. Datos del entrenamiento 5

Número de Imágenes	507
Batch	8
Subdivisiones	2
Tamaño de imágenes	256 * 256
Canales	3

Fuente: Elaboración propia

Figura 24. Estadísticas del entrenamiento 5

```
IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.920209, or 92.02 %
Total Detection Time: 2 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.5) = 0.920209
New best mAP!
Saving weights to backup//custom-yolov4-detector_best.weights
Saving weights to backup//custom-yolov4-detector_2000.weights
Saving weights to backup//custom-yolov4-detector_last.weights
Saving weights to backup//custom-yolov4-detector_final.weights
```

Fuente: Elaboración propia

El entrenamiento 5 cuenta con una precisión del 92.02%, el cual el reconocimiento del ganado bovino es similar al del entrenamiento 0, siendo muy deficiente.

Figura 25. Resultados preliminares del entrenamiento 5

Fuente: Elaboración propia

Luego se realizó el entrenamiento 6 con imágenes unificadas de todas las datas, a continuación, se muestran los parámetros:

Tabla 7. Datos del entrenamiento 6

Número de Imágenes	1630
Batch	8
Subdivisiones	2
Tamaño de imágenes	256 * 256
Canales	3

Fuente: Elaboración propia

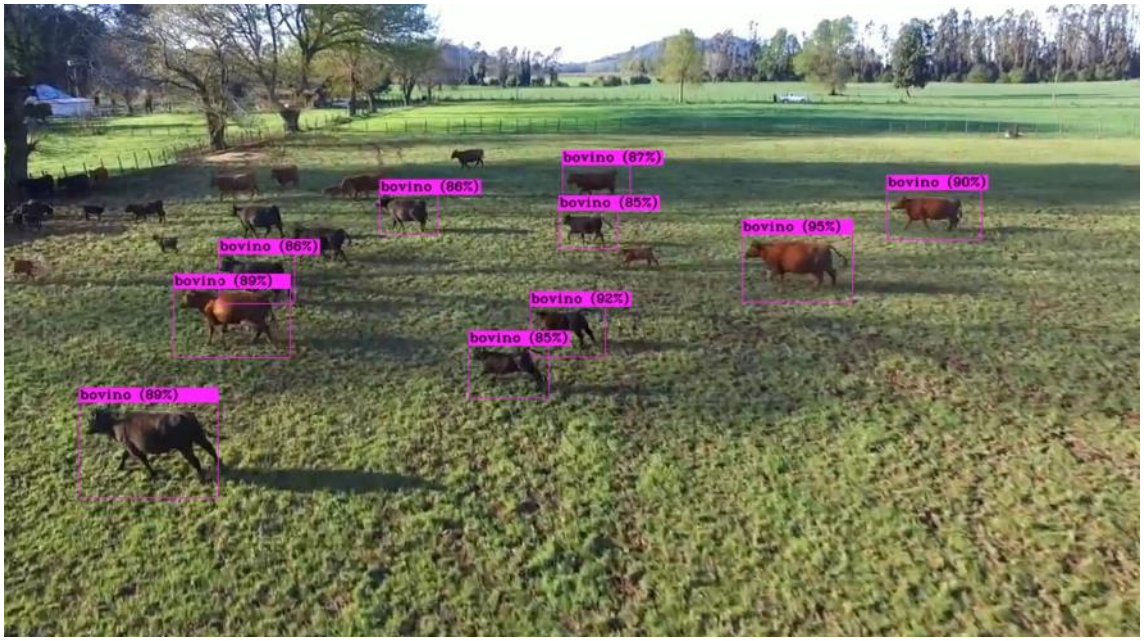
Figura 26. Estadísticas del entrenamiento 6

```
mean average precision (mAP@0.50) = 0.649962, or 65.00 %
Total Detection Time: 7 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.5) = 0.649962
New best mAP!
Saving weights to backup//custom-yolov4-detector_best.weights
Saving weights to backup//custom-yolov4-detector_2000.weights
Saving weights to backup//custom-yolov4-detector_last.weights
Saving weights to backup//custom-yolov4-detector_final.weights
```

Fuente: Elaboración propia

El entrenamiento 6 cuenta con una precisión del 65.00%, el cual el reconocimiento del ganado bovino es similar al del entrenamiento 3, pero aún existen bovinos sin ser reconocidos.

Figura 27. Resultados preliminares del entrenamiento 6



Fuente: Elaboración propia

Luego se realizó el entrenamiento 7 con imágenes unificadas de todas las datas, a continuación, se muestran los parámetros:

Tabla 8. Datos del entrenamiento 7

Número de Imágenes	1123
Batch	8
Subdivisiones	2
Tamaño de imágenes	256 * 256
Canales	3

Fuente: Elaboración propia

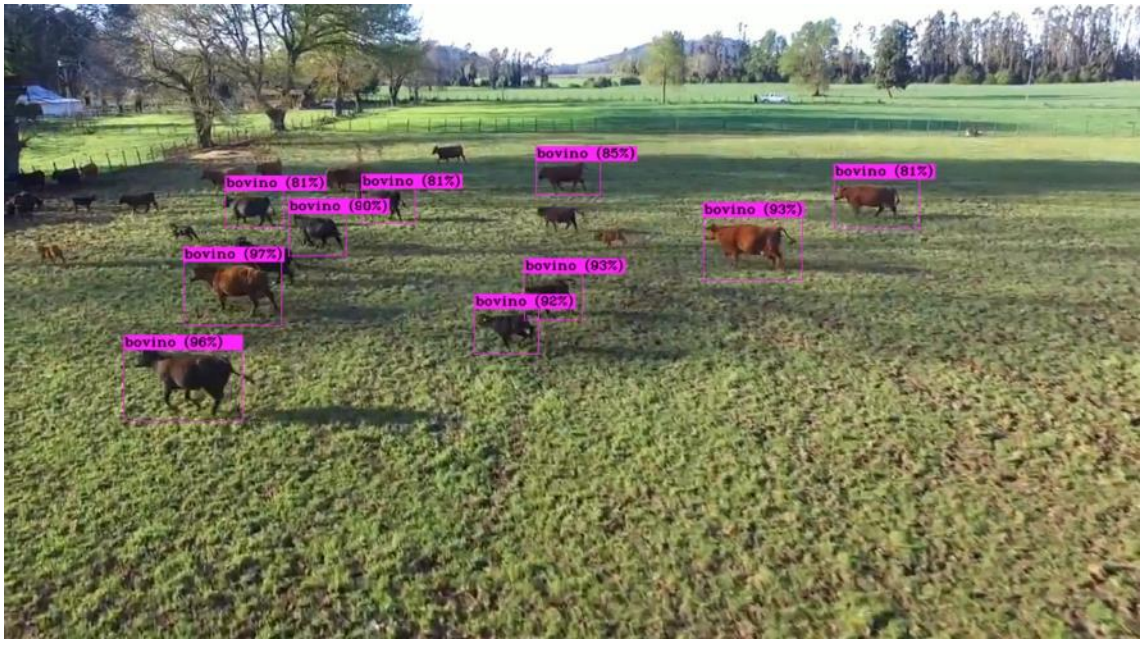
Figura 28. Estadísticas del entrenamiento 7

```
 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.619897, or 61.99 %
Total Detection Time: 5 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.5) = 0.619897
New best mAP!
Saving weights to backup//custom-yolov4-detector_best.weights
Saving weights to backup//custom-yolov4-detector_2000.weights
Saving weights to backup//custom-yolov4-detector_last.weights
Saving weights to backup//custom-yolov4-detector_final.weights
```

Fuente: Elaboración propia

El entrenamiento 6 cuenta con una precisión del 61.99%, el cual el reconocimiento del ganado bovino es similar al del entrenamiento 4, pero aún existen bovinos sin ser reconocidos.

Figura 29. Resultados preliminares del entrenamiento 7



Fuente: Elaboración propia

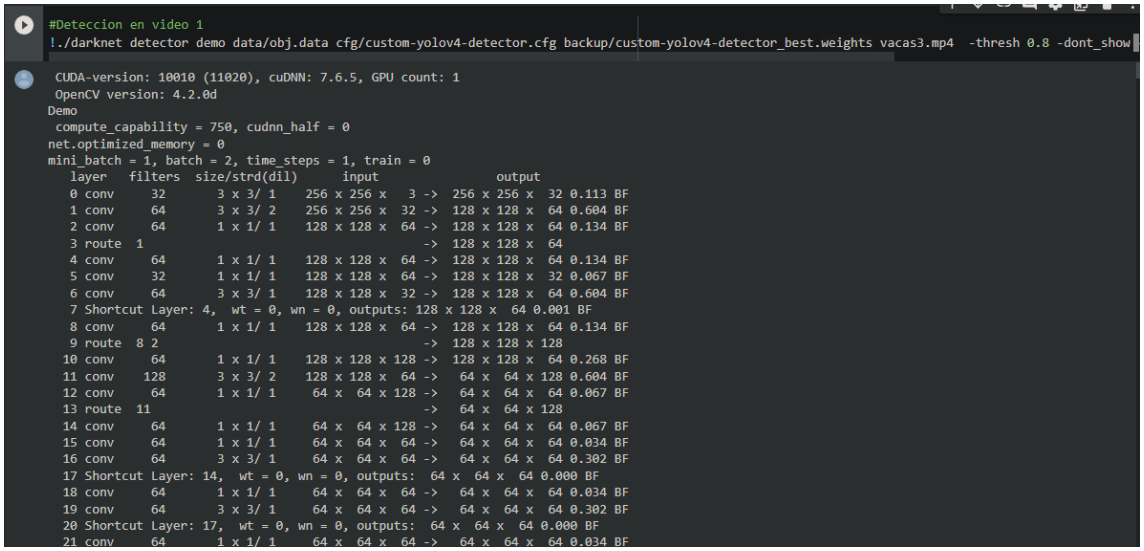
2.6. Ejecución y/o ensamblaje del prototipo

Para poner en marcha la detección del modelo se ejecuta el siguiente script dentro de la carpeta darknet del servidor Ubuntu donde “detector” sirve para indicar al modelo que realice el proceso de detección, “data/obj.data” indica los

etiquetas de las clases entrenadas, “cfg/custom-yolov4-detector.cfg” son los pesos descargados de Yolo V4, “backup/custom-yolov4-detector_best.weights” indica la ruta de los pesos del modelo entrenado, luego sigue la ruta u origen del video de entrada, el parámetro “-tresh” permite cambiar el umbral de detecciones de yolo, “-dont show” evita que se muestre el video en tiempo de detección y “-out_filename” permite escribir un nombre para el video de salida con los objetos detectados por Yolo v4:

```
./darknet detector demo data/obj.data cfg/custom-yolov4-detector.cfg
backup/custom-yolov4-detector_best.weights nombre-origen-del-video.mp4
-thresh 0.8 -dont_show -out_filename salida-video-resultado.mp4
```

Figura 30. Script de detección de bovinos con Yolo v4



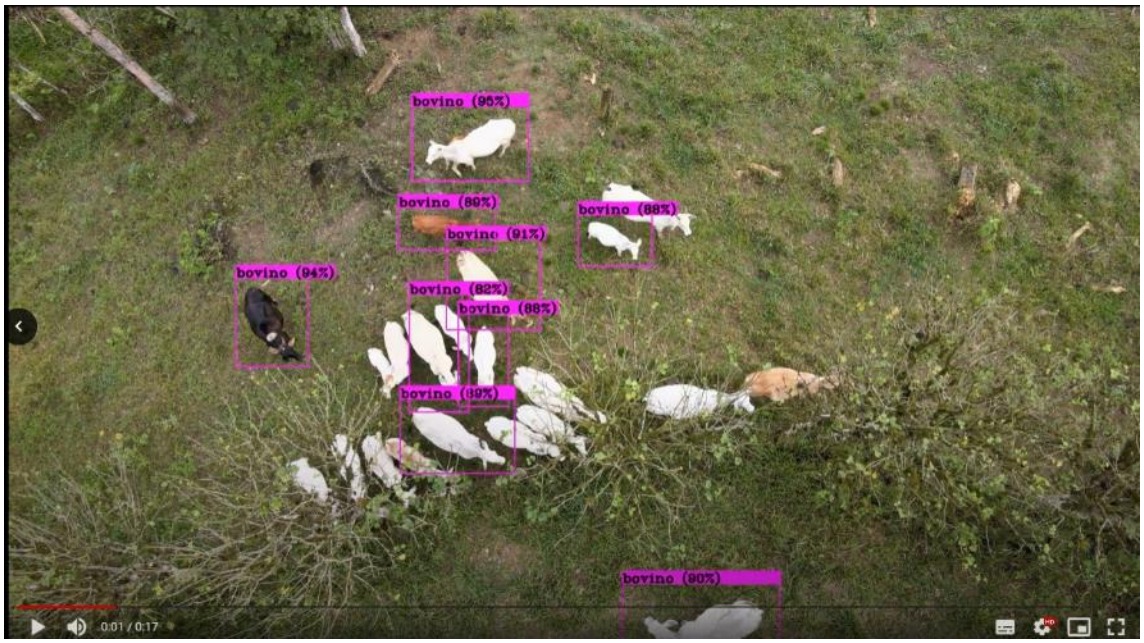
Fuente: Elaboración propia

Figura 31. Resultados de la detección

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, cls_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 22.556
avg_outputs = 185478
 Allocate additional workspace_size = 52.43 MB
Loading weights from backup/custom-yolov4-detector_best.weights...
 seen 64, trained: 13 K-images (0 Kilo-batches_64)
Done! Loaded 162 layers from weights-file
video file: vacas3.mp4
Video stream: 3840 x 2160
OpenCV: FFMPEG: tag 0x58564944/'DIVX' is not supported with codec id 12 and format 'mp4 / MP4 (MPEG-4 Part 14)'
OpenCV: FFMPEG: fallback to use tag 0x7634706d/'mp4v'
Objects:


FPS:0.0          AVG_FPS:0.0
Objects:

bovino: 95%
bovino: 91%
bovino: 89%
bovino: 89%
bovino: 89%
bovino: 88%
bovino: 87%
bovino: 82%
bovino: 82%
bovino: 80%
```

Fuente: Elaboración propia

Figura 32. Video resultante de la detección de bovinos



Fuente: Elaboración propia

3. CAPÍTULO III. EVALUACIÓN DEL MODELO

3.1. Plan de evaluación

Para el plan de evaluación de los resultados obtenidos del entrenamiento de la red neuronal se utilizaron las siguientes métricas de clasificación y de detección.

3.1.1. Prueba de Entrenamiento y Validación de la Red Neuronal

3.1.1.1. Métricas de clasificación

Para evaluar la clasificación del resultado de salida obtenida de una red neuronal se califica de forma binaria. Los valores obtenidos son verdadero positivo que indica la cantidad de detecciones correctas, el falso negativo indica la cantidad de objetos no detectados, el falso positivo indica la cantidad de objetos detectados que no forman parte de la clase de objetos entrenados por la red neuronal y el verdadero negativo indica cuando en el cuadro delimitador no aparece un objeto que ha si ha sido entrenado por la CNN y no es identificado como verdad [45].

Tabla 9. Matriz de confusión

		Predicción de la Red Neuronal	
		Objeto Presente	Objeto no Presente
Situación Real	Objeto Presente	Verdadero Positivo	Falso Negativo
	Objeto no Presente	Falso Positivo	Verdadero Negativo

Fuente: Elaboración propia

Precisión (Precision)

La métrica de precisión permite visualizar aquellos elementos seleccionados dentro de los cuadros delimitadores en la predicción sean exactos o relevantes. El cálculo se lo obtiene dividiendo el número de predicciones verdaderas positivas entre la suma de verdades positivas y falsos positivos [46].

$$Precisión = \frac{Verdadero\ Positivo}{Verdadero\ Positivo + Falso\ Positivo}$$

Sensibilidad (Recall)

La métrica de sensibilidad permite la visualización del número de objetos que son seleccionados dentro de los cuadros delimitadores. El cálculo se lo obtiene de la división de la cantidad de predicciones verdaderos positivos entre la suma de las predicciones verdaderos positivos y falsos negativos [46].

$$Sensibilidad = \frac{Verdadero\ Positivo}{Verdadero\ Positivo + Falso\ Negativo}$$

Valor de Referencia (F-Score)

El F-Score permite visualizar una referencia que tan bien va el funcionamiento del sistema, este parámetro se lo calcula de la división de la multiplicación del valor de la precisión por el valor de la sensibilidad entre la suma de ambos valores [46].

$$Valor\ de\ Referencia = \frac{Precisión \times Sensibilidad}{Precisión + Sensibilidad}$$

3.1.1.2. Métricas de detección

Las variables utilizadas para la obtención de estas métricas son la verdad de fondo y las cajas delimitadoras pronosticas de entrada.

Intersección sobre Unión (Intersection over Union)

Esta métrica es muy usada en la detección de objetos y es compatible con la mayoría de detectores utilizados. Se obtiene de la división del área de intersección entre el cuadro de predicción y la multiplicación del cuadro delimitador de la verdad por su área de unión [47].

$$IoU = \frac{Área\ de\ Intersección}{Área\ de\ Unión}$$

Media de Precisión Promedio (mAP)

Es una métrica de precisión que se encuentra de forma predeterminada dentro del conjunto de datos de PascalVOC, MSCOCO, AP50 [42]. Puede ser utilizada dentro de un conjunto de datos personalizados, este valor es calculado desde el terminal del sistema operativo en la ruta del directorio donde se encuentra el

proyecto con “darknet detector map”, seguido del archivo data, cfg y los pesos del entrenamiento.

```
./darknet detector map data/obj.data yolo-obj.cfg backup/yolo-obj_1000.weights
```

A mayor porcentaje de precisión, mayor será los resultados de detección del entrenamiento.

Promedio de Pérdida (Avg Loss)

Este valor indica el porcentaje promedio de inexactitud en la detección de objetos, a diferencia del promedio de precisión este valor debe ser mínimo para considerar el funcionamiento óptimo de la red neuronal. Este resultado es obtenido durante el entrenamiento de la red neuronal.

3.2. Resultado de evaluación

3.2.1. Resultado de la Prueba de Entrenamiento y Validación

Resultado de la prueba 0

Tabla 10. Resultados de la prueba de entrenamiento y validación 0

Parámetros	Valor
Conjunto de datos	Propio
Imágenes de entrenamiento	239
Imágenes de prueba	13
Taza de aprendizaje (Learning rate)	0.001
Promedio de pérdida (avg loss)	2.158909
Pesos pre-entrenados	yolov4.conv.137
Iteraciones	2000
Tamaño de las imágenes (alto x ancho)	256 x 256

Fuente: Elaboración propia

Tabla 11. Métricas de la prueba 0

Métricas	Valor
Precisión (Precision)	0.69
Sensibilidad (Recall)	0.58
Valor de referencia (F-Score)	0.63
Intersección sobre Unión (Intersection over Union)	47.21%
mAP	43.60%

Fuente: Elaboración propia

Los resultados obtenidos del entrenamiento 0 en la precisión es de un 0.69 que es aceptable, la sensibilidad es de un 0.58 que es aceptable, el valor de referencia es de un 0.63 que es aceptable, la intersección sobre unión es de un 47.21% que no es óptimo y el valor de la media de precisión es de un 43.60% que no es óptimo. El resultado de la detección se puede visualizar en la figura, donde se evidencia que los cuadros delimitadores no llegan a cubrir la totalidad del objeto, además solo reconoce los bovinos de piel blanca y no reconoce los demás bovinos de diferente color.

Figura 33. Resultados de detección de la prueba de entrenamiento y validación 0

Fuente: Elaboración propia

Resultado de la prueba 1

Tabla 12. Resultados de la prueba de entrenamiento 1

Parámetros	Valor
Conjunto de datos	Propio
Imágenes de entrenamiento	362
Imágenes de prueba	21
Taza de aprendizaje (Learning rate)	0.001
Promedio de pérdida (avg loss)	3.025655
Pesos pre-entrenados	yolov4.conv.137
Iteraciones	2000
Tamaño de las imágenes (alto x ancho)	256 x 256

Fuente: Elaboración propia

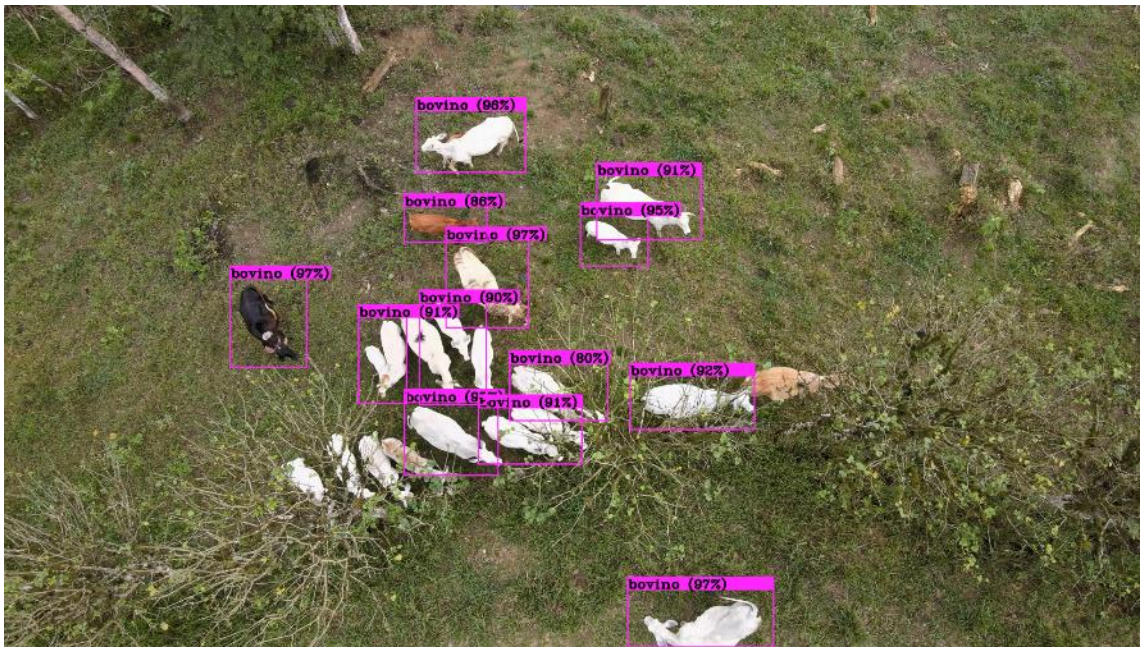
Tabla 13. Métricas de la prueba 1

Métricas	Valor
Precisión (Precision)	0.90
Sensibilidad (Recall)	0.93
Valor de referencia (F-Score)	0.92
Intersección sobre Unión (Intersection over Union)	71.61%
mAP	92.95%

Fuente: Elaboración propia

Los resultados obtenidos del entrenamiento 1 en la precisión es de un 0.90 que es óptimo, la sensibilidad es de un 0.93 que es óptimo, el valor de referencia es de un 0.92 que es óptimo, la intersección sobre unión es de un 71.61% que es aceptable y el valor de la media de precisión es de un 92.95% que es óptimo. El resultado de la detección se puede visualizar en la figura, donde se evidencia que los cuadros delimitadores llegan a cubrir la totalidad del objeto, además ya logra reconocer los bovinos de piel blanca y de diferente color. La detección también mejora.

Figura 34. Resultados de detección de la prueba de entrenamiento y validación 1



Fuente: Elaboración propia

Resultado de la prueba 2

Tabla 14. Resultados de la prueba de entrenamiento y validación 2

Parámetros	Valor
Conjunto de datos	Propio
Imágenes de entrenamiento	600
Imágenes de prueba	35
Taza de aprendizaje (Learning rate)	0.001
Promedio de pérdida (avg loss)	3.274151
Pesos pre-entrenados	yolov4.conv.137
Iteraciones	2000
Tamaño de las imágenes (alto x ancho)	256 x 256

Fuente: Elaboración propia

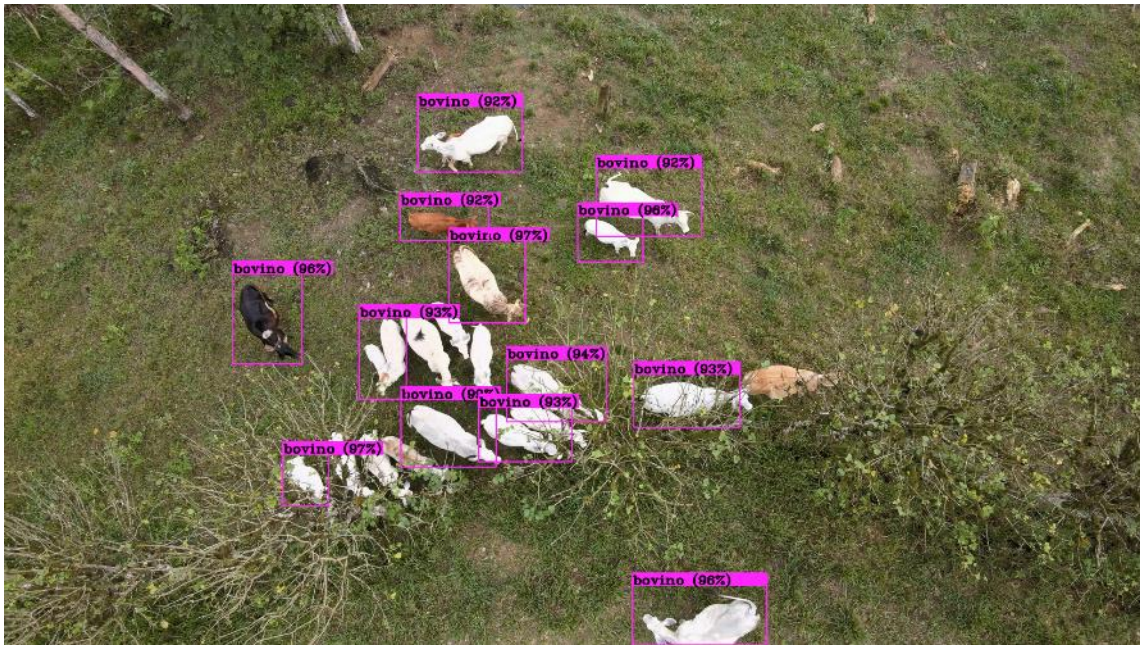
Tabla 15. Métricas de la prueba de entrenamiento y validación 2

Métricas	Valor
Precisión (Precision)	0.75
Sensibilidad (Recall)	0.70
Valor de referencia (F-Score)	0.72
Intersección sobre Unión (Intersection over Union)	55.41%
mAP	59.07%

Fuente: Elaboración propia

Los resultados obtenidos del entrenamiento 2 en la precisión es de un 0.75 que es bueno, la sensibilidad es de un 0.70 que es aceptable, el valor de referencia es de un 0.72 que es aceptable, la intersección sobre unión es de un 55.41% que es aceptable y el valor de la media de precisión es de un 59.07% que es aceptable. El resultado de la detección se puede visualizar en la figura, donde se evidencia que los cuadros delimitadores llegan a cubrir la totalidad del objeto, además la detección mejora ligeramente a la del entrenamiento 1.

Figura 35. Resultados de la prueba de entrenamiento y validación 2



Fuente: Elaboración propia

Resultado de la prueba 3

Tabla 16. Resultados de la prueba de entrenamiento 3

Parámetros	Valor
Conjunto de datos	Propio
Imágenes de entrenamiento	356
Imágenes de prueba	20
Taza de aprendizaje (Learning rate)	0.001
Promedio de pérdida (avg loss)	8.260370
Pesos pre-entrenados	yolov4.conv.137
Iteraciones	2000
Tamaño de las imágenes (alto x ancho)	256 x 256

Fuente: Elaboración propia

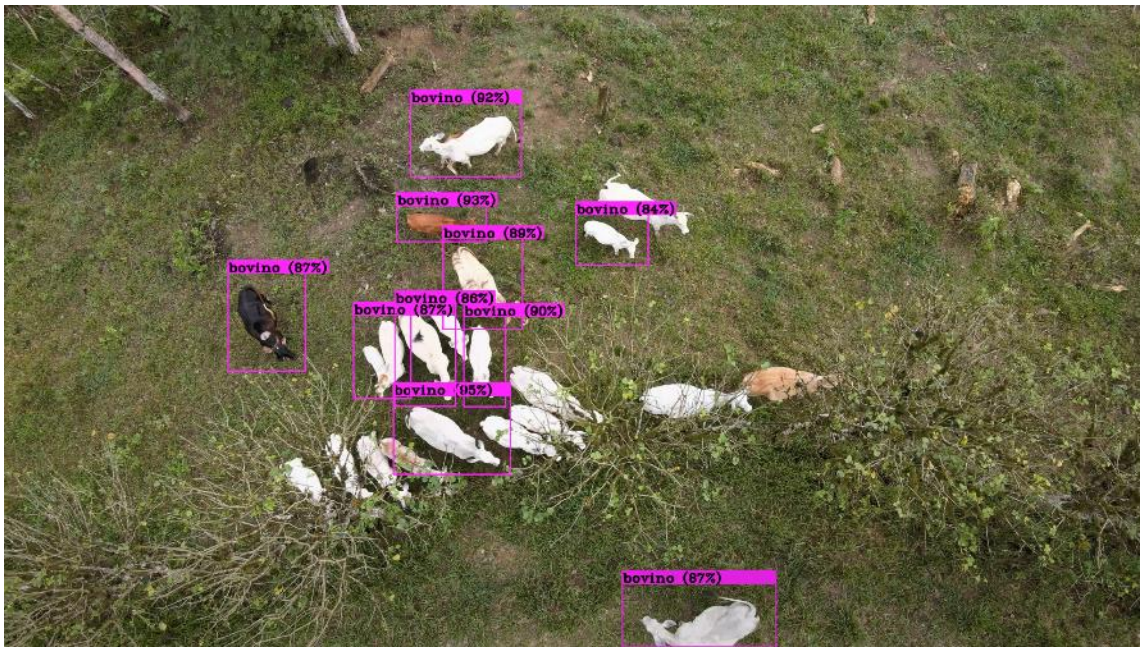
Tabla 17. Métricas de las pruebas de entrenamiento 3

Métricas	Valor
Precisión (Precision)	0.82
Sensibilidad (Recall)	0.93
Valor de referencia (F-Score)	0.87
Intersección sobre Unión (Intersection over Union)	63.50%
mAP	89.52%

Fuente: Elaboración propia

Los resultados obtenidos del entrenamiento 3 en la precisión es de un 0.82 que es bueno, la sensibilidad es de un 0.93 que es óptimo, el valor de referencia es de un 0.87 que es bueno, la intersección sobre unión es de un 63.50% que es aceptable y el valor de la media de precisión es de un 89.52% que es bueno. El resultado de la detección se puede visualizar en la figura, donde se evidencia que los cuadros delimitadores llegan a cubrir la totalidad del objeto, además la detección empeora significativamente a la del entrenamiento 1 y 2.

Figura 36. Resultados de la detección de las pruebas de entrenamiento 3



Fuente: Elaboración propia

Resultado de la prueba 4

Tabla 18. Resultados de las pruebas de entrenamiento y validación 4

Parámetros	Valor
Conjunto de datos	Propio
Imágenes de entrenamiento	717
Imágenes de prueba	42
Taza de aprendizaje (Learning rate)	0.001
Promedio de pérdida (avg loss)	6.186623
Pesos pre-entrenados	yolov4.conv.137
Iteraciones	2000
Tamaño de las imágenes (alto x ancho)	256 x 256

Fuente: Elaboración propia

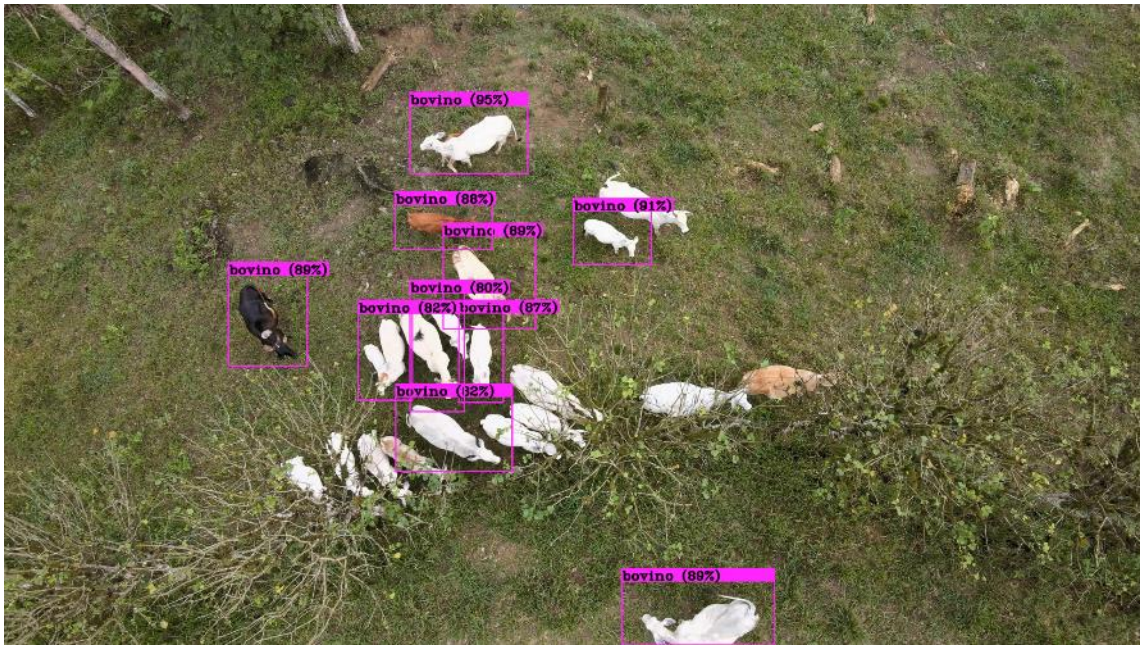
Tabla 19. Métricas del entrenamiento 4

Métricas	Valor
Precisión (Precision)	0.88
Sensibilidad (Recall)	0.91
Valor de referencia (F-Score)	0.89
Intersección sobre Unión (Intersection over Union)	67.71%
mAP	90.32%

Fuente: Elaboración propia

Los resultados obtenidos del entrenamiento 4 en la precisión es de un 0.88 que es bueno, la sensibilidad es de un 0.91 que es óptimo, el valor de referencia es de un 0.89 que es bueno, la intersección sobre unión es de un 67.71% que es aceptable y el valor de la media de precisión es de un 90.32% que es óptimo. El resultado de la detección se puede visualizar en la figura, donde se evidencia que los cuadros delimitadores llegan a cubrir la totalidad del objeto, además la detección es similar a la del entrenamiento 3.

Figura 37. Resultados de detección del entrenamiento 4



Fuente: Elaboración propia

Resultado de la prueba 5

Tabla 20. Resultados de las pruebas de entrenamiento y validación 5

Parámetros	Valor
Conjunto de datos	Propio
Imágenes de entrenamiento	507
Imágenes de prueba	25
Taza de aprendizaje (Learning rate)	0.001
Promedio de pérdida (avg loss)	2.442720
Pesos pre-entrenados	yolov4.conv.137
Iteraciones	2000
Tamaño de las imágenes (alto x ancho)	256 x 256

Fuente: Elaboración propia

Tabla 21. Métricas de las pruebas de entrenamiento 5

Métricas	Valor
Precisión (Precision)	0.83
Sensibilidad (Recall)	0.94
Valor de referencia (F-Score)	0.88
Intersección sobre Unión (Intersection over Union)	64.90%
mAP	92.02%

Fuente: Elaboración propia

Los resultados obtenidos del entrenamiento 5 en la precisión es de un 0.83 que es bueno, la sensibilidad es de un 0.94 que es óptimo, el valor de referencia es de un 0.88 que es bueno, la intersección sobre unión es de un 64.90% que es aceptable y el valor de la media de precisión es de un 90.32% que es óptimo. El resultado de la detección se puede visualizar en la figura, donde se evidencia que los cuadros delimitadores llegan a cubrir la totalidad del objeto, además la detección empeora significativamente en comparación con los entrenamientos anteriores.

Figura 38. Resultados de la detección de la prueba de entrenamiento 5

Fuente: Elaboración propia

Resultado de la prueba 6

Tabla 22. Resultados de la prueba de entrenamiento y validación 6

Parámetros	Valor
Conjunto de datos	Propio
Imágenes de entrenamiento	1386
Imágenes de prueba	81
Taza de aprendizaje (Learning rate)	0.001
Promedio de pérdida (avg loss)	4.997972
Pesos pre-entrenados	yolov4.conv.137
Iteraciones	2000
Tamaño de las imágenes (alto x ancho)	256 x 256

Fuente: Elaboración propia

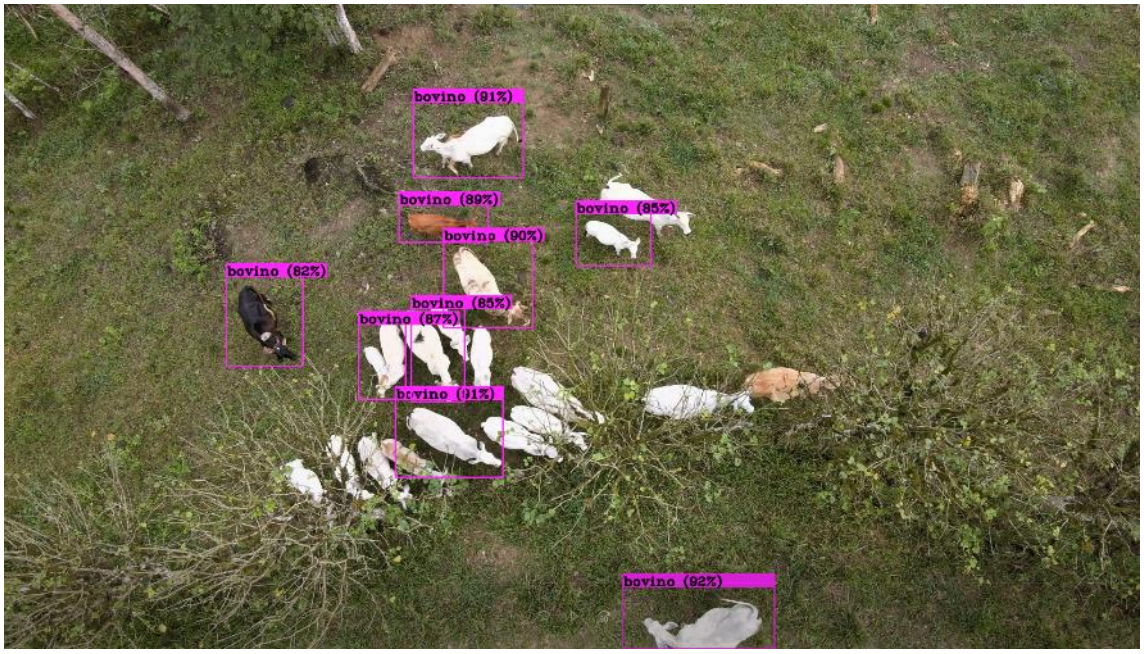
Tabla 23. Métricas de la prueba de entrenamiento 6

Métricas	Valor
Precisión (Precision)	0.75
Sensibilidad (Recall)	0.74
Valor de referencia (F-Score)	0.75
Intersección sobre Unión (Intersection over Union)	56.12%
mAP	65.00%

Fuente: Elaboración propia

Los resultados obtenidos del entrenamiento 6 en la precisión es de un 0.75 que es bueno, la sensibilidad es de un 0.74 que es aceptable, el valor de referencia es de un 0.75 que es bueno, la intersección sobre unión es de un 56.12% que es aceptable y el valor de la media de precisión es de un 65.00% que es aceptable. El resultado de la detección se puede visualizar en la figura, donde se evidencia que los cuadros delimitadores llegan a cubrir la totalidad del objeto, además la detección es similar a los entrenamientos 3 y 4.

Figura 39. Resultado de la detección de la prueba de entrenamiento 6



Fuente: Elaboración propia

Resultado de la prueba 7

Tabla 24. Resultados de la prueba de entrenamiento y validación 7

Parámetros	Valor
Conjunto de datos	Propio
Imágenes de entrenamiento	955
Imágenes de prueba	56
Taza de aprendizaje (Learning rate)	0.001
Promedio de pérdida (avg loss)	6.003450
Pesos pre-entrenados	yolov4.conv.137
Iteraciones	2000
Tamaño de las imágenes (alto x ancho)	256 x 256

Fuente: Elaboración propia

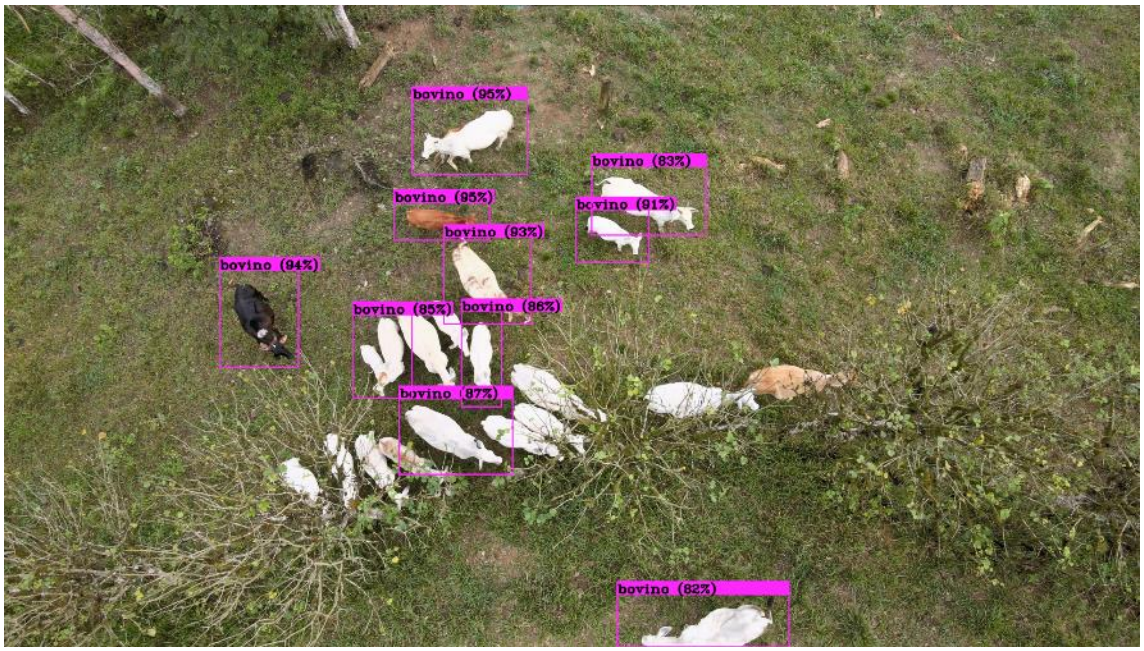
Tabla 25. Métricas de la prueba de entrenamiento 7

Métricas	Valor
Precisión (Precision)	0.75
Sensibilidad (Recall)	0.73
Valor de referencia (F-Score)	0.74
Intersección sobre Unión (Intersection over Union)	55.77%
mAP	61.99%

Fuente: Elaboración propia

Los resultados obtenidos del entrenamiento 7 en la precisión es de un 0.75 que es bueno, la sensibilidad es de un 0.73 que es aceptable, el valor de referencia es de un 0.74 que es aceptable, la intersección sobre unión es de un 55.77% que es aceptable y el valor de la media de precisión es de un 61.99% que es aceptable. El resultado de la detección se puede visualizar en la figura, donde se evidencia que los cuadros delimitadores llegan a cubrir la totalidad del objeto, además la detección es similar a los entrenamientos 4 y 6.

Figura 40. Resultados de la detección de la prueba de entrenamiento 7



Fuente: Elaboración propia

3.2.2. Resultado de la Prueba de Rendimiento

Resultado de la prueba de rendimiento 0.

Figura 41. Resultados de la detección de la prueba de rendimiento 0

Fuente: Elaboración propia

Tabla 26. Información del video de la prueba de rendimiento 0

Información del Video		
Video	vacas1_detectado.mp4	
Duración	51 segundos	
Detección	**Bovino**	**Nivel de detección**
Detección de la red	5	0.138888889
Conteo manual	36	

Fuente: Elaboración propia

Tabla 27. Matriz de confusión de la prueba de rendimiento 0

Matriz de Confusión			
		Resultado de Predicción	
		Bovino	**No Bovino**
Situación Real	**Bovino**	5	31
	No Bovino	0	0
Cálculo de Resultados			
		Precisión	1.00
		Sensibilidad	0.14
		Valor de Referencia	0.25
		mAP	13.89%

Fuente: Elaboración propia

Para esta prueba se usó un vídeo tomado desde un dron con una cámara a 40 metros de altura, la red neuronal fue entrenada con 239 imágenes y la detección fue realizada a través de una laptop con GPU NVIDIA para que los resultados fueran rápidos en procesarse; los resultados obtenidos de esta prueba no fueron óptimos en la detección de bovinos.

Resultado de la prueba de rendimiento 1.

Figura 42. Resultados de la detección de la prueba de rendimiento 1

Fuente: Elaboración propia

Tabla 28. Información del video de la prueba de rendimiento 1

Información del Video		
Video	vacas1_1_detectado.mp4	
Duración	51 segundos	
Detección	**Bovino**	**Nivel de detección**
Detección de la red	24	0.666666667
Conteo manual	36	

Fuente: Elaboración propia

Tabla 29. Matriz de confusión de la prueba de rendimiento 1

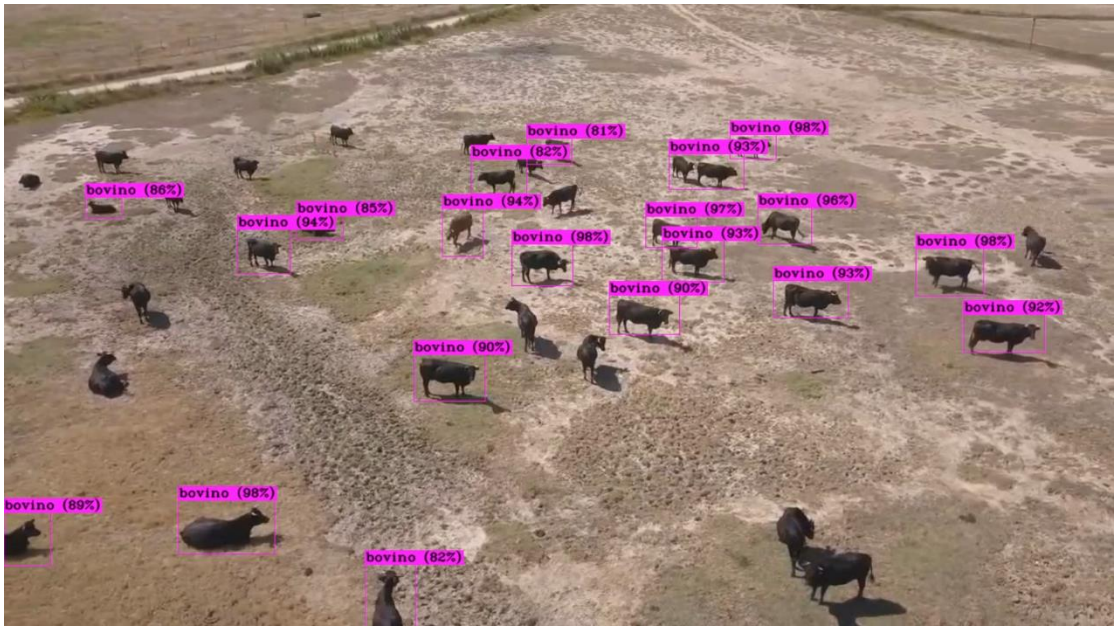
Matriz de Confusión			
		Resultado de Predicción	
		Bovino	**No Bovino**
Situación Real	**Bovino**	24	12
	No Bovino	0	0
Cálculo de Resultados			
		Precisión	1.00
		Sensibilidad	0.67
		Valor de Referencia	0.80
		mAP	66.67%

Fuente: Elaboración propia

Para esta prueba se usó un vídeo tomado desde un dron con una cámara a 40 metros de altura, la red neuronal fue entrenada con 362 imágenes y la detección fue realizada a través de una laptop con GPU NVIDIA para que los resultados fueran rápidos en procesarse; los resultados obtenidos de esta prueba fueron óptimos en la detección de bovinos.

Resultado de la prueba de rendimiento 2.

Figura 43. Resultados de la prueba de rendimiento 2



Fuente: Elaboración propia

Tabla 30. Información del video de la prueba de rendimiento 2

Información del Video		
Video	vacas1_1_2_detectado.mp4	
Duración	51 segundos	
Detección	**Bovino**	**Nivel de detección**
Detección de la red	21	0.583333333
Conteo manual	36	

Fuente: Elaboración propia

Tabla 31. Matriz de confusión de la prueba de rendimiento 2

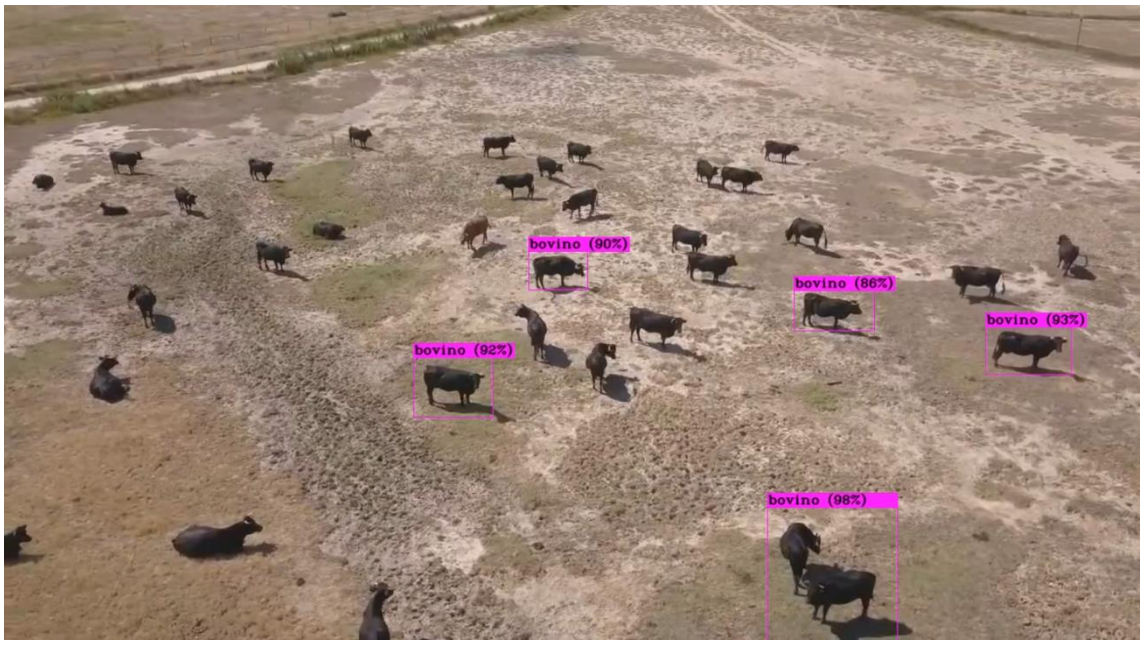
Matriz de Confusión			
		Resultado de Predicción	
		Bovino	**No Bovino**
Situación Real	**Bovino**	21	15
	No Bovino	0	0
Cálculo de Resultados			
		Precisión	1.00
		Sensibilidad	0.58
		Valor de Referencia	0.73
		mAP	58.33%

Fuente: Elaboración propia

Para esta prueba se usó un vídeo tomado desde un dron con una cámara a 40 metros de altura, la red neuronal fue entrenada con 600 imágenes y la detección fue realizada a través de una laptop con GPU NVIDIA para que los resultados fueran rápidos en procesarse; los resultados obtenidos de esta prueba fueron buenos en la detección de bovinos.

Resultado de la prueba de rendimiento 3.

Figura 44. Resultados de detección de la prueba de rendimiento 3



Fuente: Elaboración propia

Tabla 32. Información del video de la prueba de rendimiento 3

Información del Video		
Video	vacas1_1_3_detectado.mp4	
Duración	51 segundos	
Detección	**Bovino**	**Nivel de detección**
Detección de la red	6	0.166666667
Conteo manual	36	

Fuente: Elaboración propia

Tabla 33. Matriz de confusión de la prueba de rendimiento 3

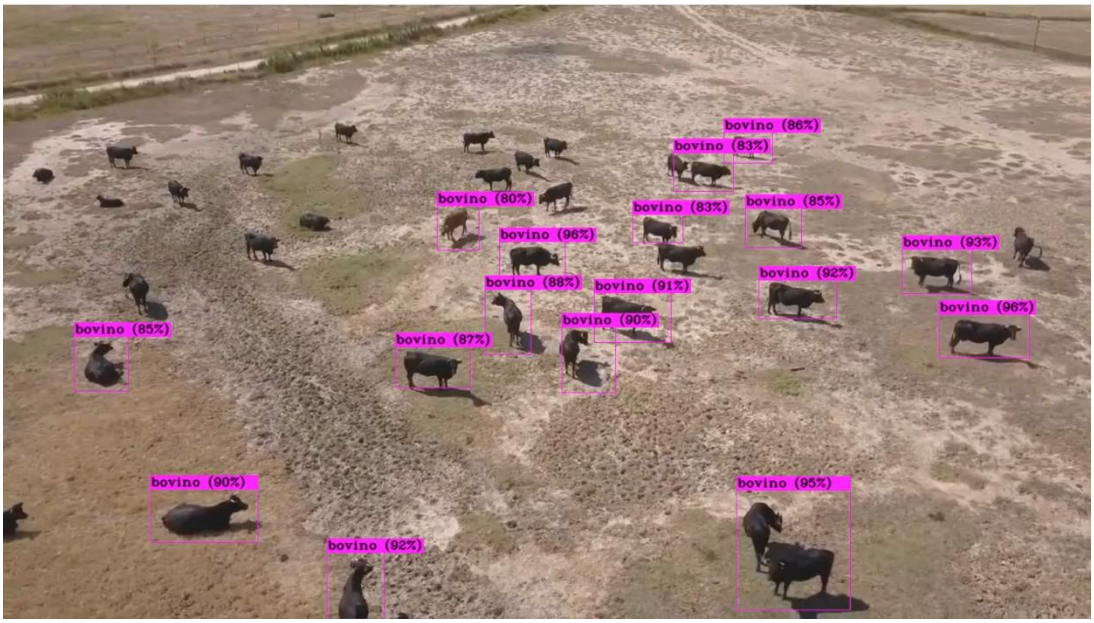
Matriz de Confusión			
		Resultado de Predicción	
		Bovino	**No Bovino**
Situación Real	**Bovino**	6	30
	No Bovino	0	0
Cálculo de Resultados			
		Precisión	1.00
		Sensibilidad	0.17
		Valor de Referencia	0.29
		mAP	16.67%

Fuente: Elaboración propia

Para esta prueba se usó un vídeo tomado desde un dron con una cámara a 40 metros de altura, la red neuronal fue entrenada con 356 imágenes y la detección fue realizada a través de una laptop con GPU NVIDIA para que los resultados fueran rápidos en procesarse; los resultados obtenidos de esta prueba no fueron óptimos en la detección de bovinos.

Resultado de la prueba de rendimiento 4.

Figura 45. Resultados de la detección de la prueba de rendimiento 4



Fuente: Elaboración propia

Tabla 34. Información del video de la prueba de rendimiento 4

Información del Video		
Video	vacas1_1_4_detectado.mp4	
Duración	51 segundos	
Detección	**Bovino**	**Nivel de detección**
Detección de la red	18	0.50
Conteo manual	36	

Fuente: Elaboración propia

Tabla 35. Matriz de confusión de la prueba de rendimiento 4

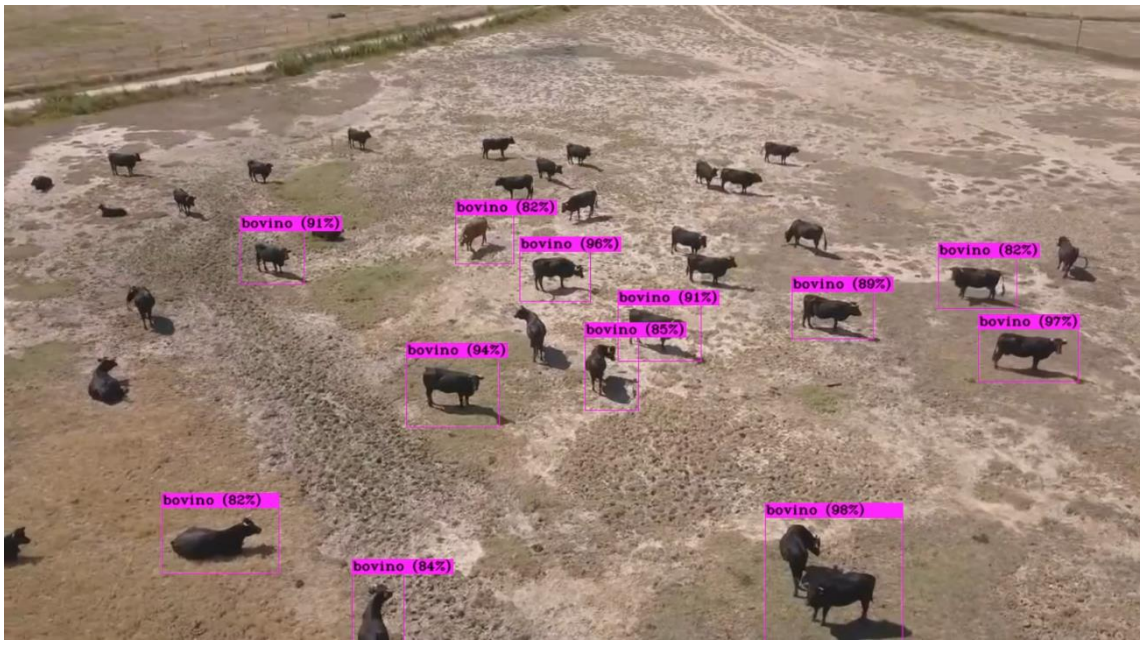
Matriz de Confusión			
		Resultado de Predicción	
		Bovino	**No Bovino**
Situación Real	**Bovino**	18	18
	No Bovino	0	0
Cálculo de Resultados			
		Precisión	1.00
		Sensibilidad	0.50
		Valor de Referencia	0.67
		mAP	50.00%

Fuente: Elaboración propia

Para esta prueba se usó un vídeo tomado desde un dron con una cámara a 40 metros de altura, la red neuronal fue entrenada con 717 imágenes y la detección fue realizada a través de una laptop con GPU NVIDIA para que los resultados fueran rápidos en procesarse; los resultados obtenidos de esta prueba fueron aceptables en la detección de bovinos.

Resultado de la prueba de rendimiento 5.

Figura 46. Resultados de la prueba de detección de la prueba de rendimiento 5



Fuente: Elaboración propia

Tabla 36. Información del video de la prueba de rendimiento 5

Información del Video		
Video	vacas1_1_5_detectado.mp4	
Duración	51 segundos	
Detección	**Bovino**	**Nivel de detección**
Detección de la red	13	0.361111111
Conteo manual	36	

Fuente: Elaboración propia

Tabla 37. Matriz de confusión de la prueba de rendimiento 5

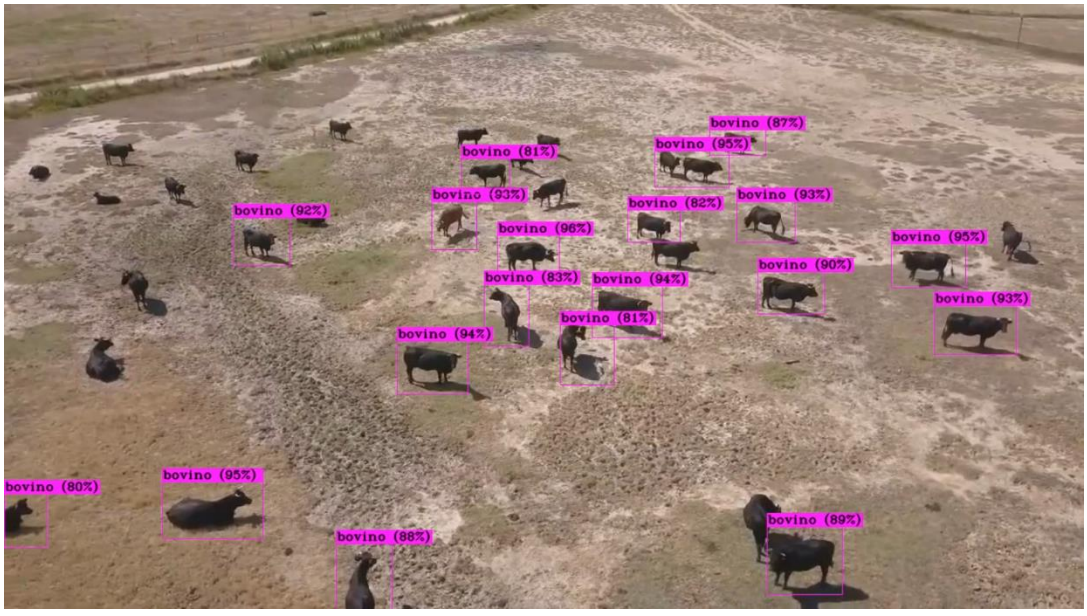
Matriz de Confusión			
		Resultado de Predicción	
		Bovino	**No Bovino**
Situación Real	**Bovino**	13	23
	No Bovino	0	0
Cálculo de Resultados			
		Precisión	1.00
		Sensibilidad	0.36
		Valor de Referencia	0.53
		mAP	36.11%

Fuente: Elaboración propia

Para esta prueba se usó un vídeo tomado desde un dron con una cámara a 40 metros de altura, la red neuronal fue entrenada con 507 imágenes y la detección fue realizada a través de una laptop con GPU NVIDIA para que los resultados fueran rápidos en procesarse; los resultados obtenidos de esta prueba fueron aceptables en la detección de bovinos.

Resultado de la prueba de rendimiento 6.

Figura 47. Resultado de detección de la prueba de rendimiento 6



Fuente: Elaboración propia

Tabla 38. Información del video de la prueba de rendimiento 6

Información del Video		
Video	vacas1_1_6_detectado.mp4	
Duración	51 segundos	
Detección	**Bovino**	**Nivel de detección**
Detección de la red	20	0.555555556
Conteo manual	36	

Fuente: Elaboración propia

Tabla 39. Matriz de confusión de la prueba de rendimiento 6

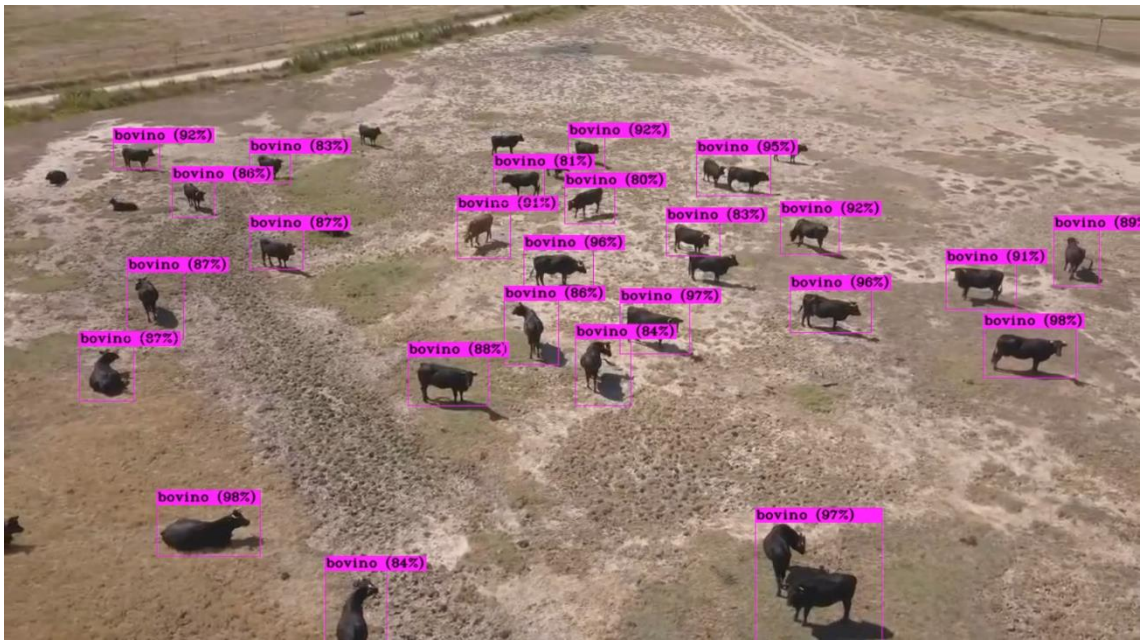
Matriz de Confusión			
		Resultado de Predicción	
		Bovino	**No Bovino**
Situación Real	**Bovino**	20	16
	No Bovino	0	0
Cálculo de Resultados			
		Precisión	1.00
		Sensibilidad	0.56
		Valor de Referencia	0.62
		mAP	55.56%

Fuente: Elaboración propia

Para esta prueba se usó un vídeo tomado desde un dron con una cámara a 40 metros de altura, la red neuronal fue entrenada con 1386 imágenes y la detección fue realizada a través de una laptop con GPU NVIDIA para que los resultados fueran rápidos en procesarse; los resultados obtenidos de esta prueba fueron aceptables en la detección de bovinos.

Resultado de la prueba de rendimiento 7.

Figura 48. Resultados de detección de la prueba de rendimiento 7



Fuente: Elaboración propia

Tabla 40. Información del video de la prueba de rendimiento 7

Información del Video		
Video	vacas1_1_7_detectado.mp4	
Duración	51 segundos	
Detección	**Bovino**	**Nivel de detección**
Detección de la red	27	0.75
Conteo manual	36	

Fuente: Elaboración propia

Tabla 41. Matriz de confusión de la prueba de rendimiento 7

Matriz de Confusión			
		Resultado de Predicción	
		Bovino	**No Bovino**
Situación Real	**Bovino**	27	9
	No Bovino	0	0
Cálculo de Resultados			
		Precisión	1.00
		Sensibilidad	0.75
		Valor de Referencia	0.86
		mAP	75.00%

Fuente: Elaboración propia

Para esta prueba se usó un vídeo tomado desde un dron con una cámara a 40 metros de altura, la red neuronal fue entrenada con 955 imágenes y la detección fue realizada a través de una laptop con GPU NVIDIA para que los resultados fueran rápidos en procesarse; los resultados obtenidos de esta prueba fueron óptimos en la detección de bovinos.

Con las pruebas realizadas se elaboró una matriz para obtener el modelo entrenado más óptimo para la detección de bovinos utilizando una escala de Likert.

Tabla 42. Escala de Likert

Escala	Puntaje
Óptimo (O)	3
Bueno (B)	2
Aceptable (A)	1
No óptimo (N)	0

Fuente: Elaboración propia

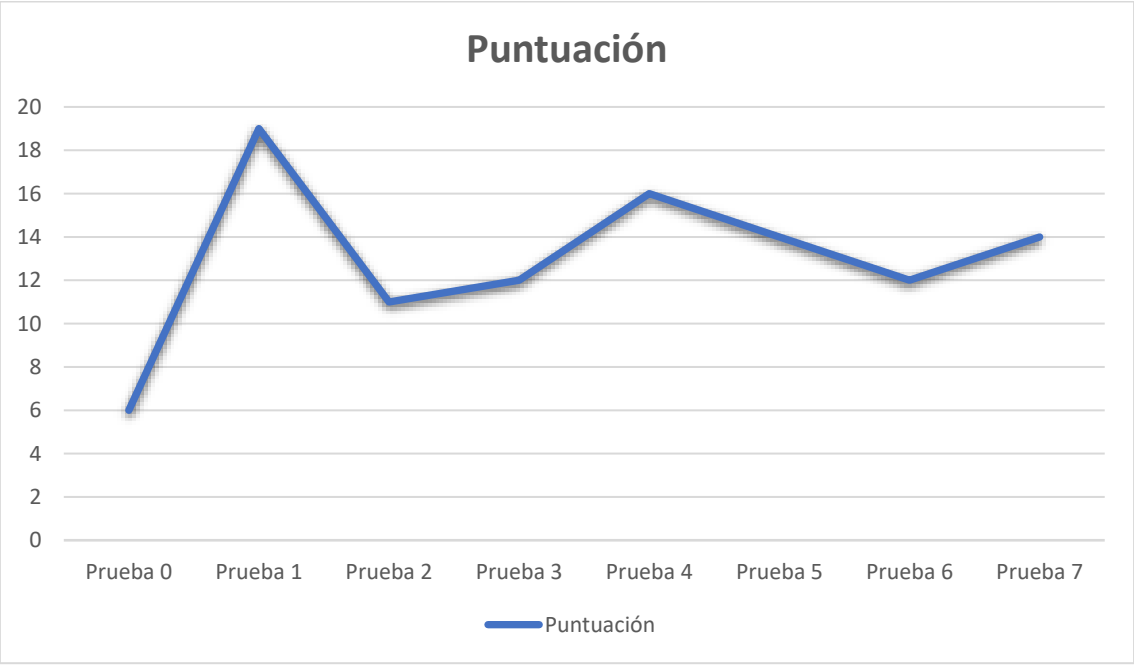
Tabla 43. Resultados de la evaluación de las pruebas de entrenamiento y rendimiento

Pruebas	Entrenamiento y validación				Rendimiento				Punt.
	Prec.	Recall	F.Score	mAP	Prec.	Recall	F.Score	mAP	Total
0	A	A	A	N	O	N	N	N	**6**
1	O	O	O	O	O	A	B	A	**19**
2	B	A	A	A	O	A	A	A	**11**
3	B	O	B	B	O	N	N	N	**12**
4	B	O	B	O	O	A	A	A	**16**
5	B	O	B	O	O	N	A	N	**14**
6	B	A	B	A	O	A	A	A	**12**
7	B	A	A	A	O	B	B	B	**14**

Fuente: Elaboración propia

Se ha elegido el modelo entrenado de la prueba 1 ya que presenta más resultados óptimos en la detección de bovinos.

Figura 49. Resultados de la evaluación de las pruebas de entrenamiento y rendimiento



Fuente: Elaboración propia

3.3. Conclusiones

Como resultado del desarrollo de un modelo de reconocimiento de ganado bovino usando deep learning se concluye que:

- Se logró desarrollar un modelo de reconocimiento de ganado bovino mediante la aplicación de algoritmos de aprendizaje profundo.
- Se seleccionó la red neuronal Yolo v4 sobre Darknet para la detección de ganado bovino mediante la utilización del Deep Learning.
- Se elaboraron 4 conjuntos de datos de diferentes ganaderías con un total de 1630 imágenes tomadas desde un dron.
- Se utilizó la herramienta LabelImg para el etiquetado de las imágenes previo al entrenamiento de los modelos.
- Se realizaron un total de 8 entrenamientos mediante la utilización de 1630 imágenes de animales bovinos.
- Se realizaron las pruebas pertinentes para seleccionar el modelo más óptimo para la detección de bovinos.

3.4. Recomendaciones

Para el desarrollo de un modelo de reconocimiento de ganado bovino usando deep learning se recomienda:

- Aplicar una exploración más profunda en fuentes bibliográficas confiables para mejorar los conocimientos adquiridos en el área del Deep Learning.
- Realizar una mayor investigación de varios modelos y técnicas de reconocimiento y detección de objetos para poder elegir el que mejor se ajuste a las necesidades de la investigación.
- Para el entrenamiento de la red neuronal se recomienda contar con un hardware que cuente con una CPU de varios núcleos con alto IPC y frecuencia y contar con una GPU con una cantidad considerable de núcleos CUDA y de por lo menos 6 GB de VRAM o superior.
- Investigar a fondo las versiones específicas de las herramientas y librerías utilizadas en la red neuronal seleccionada con el fin de evitar conflictos y errores.

- Las imágenes utilizadas para el entrenamiento deben ser claras, nítidas, de una buena resolución y tamaño para obtener mejores resultados en los entrenamientos.

4. BIBLIOGRAFIA

[1] E. Sonwani, U. Bansal, R. Alroobaea, A. M. Baqasah y M. Hedabou, «An Artificial Intelligence Approach Toward Food Spoilage Detection and Analysis,» *Frontiers in Public Health,* p. 2254, 2022.

[2] M. Viljanen y H. Parviainen, «AI Applications and Regulation: Mapping the Regulatory Strata,» *Frontiers in Computer Science,* vol. III, p. 141, 2022.

[3] J. C. Rocha, F. J. Passalia, F. D. Matos, M. B. Takahashi, D. d. S. Ciniciato, M. P. Maserati, M. F. Alves, T. Guibu de Almeida, B. L. Cardoso, A. C. Basso y M. F. Gouveia Nogueira, «A Method Based on Artificial Intelligence To Fully Automatize The Evaluation of Bovine Blastocyst Images,» *Scientific Reports,* vol. VII, nº 1, pp. 1-10, 2017.

[4] I. Vågsholm, S. N. Arzoomand y S. Boqvist, «Food Security, Safety, and Sustainability—Getting the Trade-Offs Right,» *Frontiers in Sustainable Food Systems,* vol. IV, p. 16, 2020.

[5] M. F. Gouveia Nogueira, V. B. Guilherme, M. Pronunciate, P. H. dos Santos, D. L. Bezerra da Silva y J. C. Rocha, «Artificial Intelligence-Based Grading Quality of Bovine Blastocyst Digital Images: Direct Capture with Juxtaposed Lenses of Smartphone Camera and Stereomicroscope Ocular Lens,» *Sensors,* vol. XVIII, nº 12, p. 4440, 2018.

[6] I. V. Levchenko y P. A. Merenkova, «Formation of content modules for teaching artificial intelligence in the basic school,» *RUDN Journal of Informatization in Education,* vol. XVIII, nº 3, pp. 227-237, 2021.

[7] N. Sandhya, N. M. Sashikumar, M. Priyanka, S. M. Wenisch y K. Kumarasamy, «Automated Fabric Defect Detection and Classification: A Deep Learning Approach,» *Textile & Leather Review,* vol. IV, pp. 315-335,

2021.

[8] K. Sharma, S. Lee-Cultura y M. Giannakos, «Keep Calm and Do Not Carry-Forward: Toward Sensor-Data Driven AI Agent to Enhance Human Learning,» *Frontiers in Artificial Intelligence,* vol. IV, p. 198, 2022.

[9] E. Mohammad, F. Panahi, A. N. Ahmed, A. H. Mosavi y A. El-Shafie, «Inclusive Multiple Model Using Hybrid Artificial Neural Networks for Predicting Evaporation,» *Frontiers in Environmental Science,* vol. IX, p. 652, 2022.

[10] K. Akshay, D. Samiayya, P. M. Vincent, K. Srinivasan, C.-Y. Chang y H. Ganesh, «A Hybrid Framework for Intrusion Detection in Healthcare Systems Using Deep Learning,» *Frontiers in Public Health,* vol. IX, p. 2295, 2022.

[11] A. Mohamed, M. Tharwat, M. Magdy, T. Abubakr, O. Nasr y M. Youssef, «DeepFeat: Robust Large-Scale Multi-Features Outdoor Localization in LTE Networks Using Deep Learning,» *IEEE Access,* vol. X, pp. 3400-3414, 2022.

[12] J. You, Q. Wang, R. Wang, Q. An, J. Wang, Z. Yuan, J. Wang, H. Chen, Z. Yan, J. Wei y W. Wang, «Deep Learning-Aided Automatic Contouring of Clinical Target Volumes for Radiotherapy in Breast Cancer After Modified Radical Mastectomy,» *Frontiers in Physics,* vol. IX, p. 794, 2022.

[13] M.-s. Kim, J. H. Cha, S. Lee, L. Han, W. Park, J. S. Ahn y S.-C. Park, «Deep-Learning-Based Cerebral Artery Semantic Segmentation in Neurosurgical Operating Microscope Vision Using Indocyanine Green Fluorescence Videoangiography,» *Frontiers in Neurorobotics,* vol. XV, p. 178, 2022.

[14] M. Sirshar, M. F. Khalil Paracha, M. U. Akram, N. S. Alghamdi, S. Z. Yousuf Zaidi y T. Fatima, «Attention based automated radiology report generation using CNN and LSTM,» *PLoS ONE,* vol. XVII, nº 1, p. e0262209, 2022.

[15] P. U. Putra, K. Shima y K. Shimatani, «A deep neural network model for

multi-view human activity recognition,» *PLoS ONE,* vol. XVII, nº 1, p. e0262181, 2022.

[16] K. Avazov, M. Mukhiddinov, F. Makhmudov y Y. Im Cho, «Fire Detection Method in Smart City Environments Using a Deep-Learning-Based Approach,» *Electronics,* vol. XI, nº 73, p. 73, 2021.

[17] Y. Nambu, T. Mariya, S. Shinkai, M. Umemoto, H. Asanuma, I. Sato, Y. Hirohashi, T. Torigoe, Y. Fujino y T. Saito, «A screening assistance system for cervical cytology of squamous cell atypia based on a two-step combined CNN algorithm with label smoothing,» *Cancer Medicine,* vol. XI, nº 2, pp. 520-529, 2022.

[18] Canonical, «Ubuntu,» 2022. [En línea]. Available: https://ubuntu.com/.

[19] J. L. Jahn, N. Bardele, J. T. Simes y B. Western, «Clustering of health burdens in solitary confinement: A mixed-methods approach,» *SSM: Qualitative Research in Health,* vol. II, p. 100036, 2022.

[20] Microsoft , «https://blogs.windows.com/,» 2022. [En línea]. Available: https://blogs.windows.com/.

[21] OpenCV team, «OpenCV,» 2022. [En línea]. Available: https://opencv.org/.

[22] Y.-H. Chang, Y.-J. Chen, R.-H. Huang y Y.-T. Yu, «Enhanced Image Captioning with Color Recognition Using Deep Learning Methods,» *Applied Sciences,* vol. XII, nº 209, p. 209, 2022.

[23] The Matplotlib Development team, «Matplotlib: Visualization with Python,» 2021. [En línea]. Available: https://matplotlib.org/.

[24] I. Ros, «Procesador: qué es y qué elementos lo forman, todo lo que debes saber,» Total Publishing Network S.A., 2018. [En línea]. Available: https://www.muycomputer.com/2018/10/24/procesador-que-es/.

[25] Intel, «About Intel,» 2022. [En línea]. Available: https://www.intc.com/about-intel.

[26] Intel, «Comet Lake S: Descripción general,» [En línea]. Available: https://www.intel.la/content/www/xl/es/products/platforms/details/comet-lake-s.html.

[27] «Tarjeta gráfica – todo lo que debes saber,» Profesional Review, 2019. [En línea]. Available: https://www.profesionalreview.com/tarjeta-grafica/.

[28] S. S. Teri, I. A. Musliman y A. A. Rahman, «GPU UTILIZATION IN GEOPROCESSING BIG GEODATA: A REVIEW,» *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences,* Vols. %1 de %2XLVI-4-W3-2021, pp. 295-304, 2022.

[29] NVIDIA Corporation, «About NVIDIA Corporation,» 2022. [En línea]. Available: https://www.nvidia.com/en-us/about-nvidia/.

[30] NVIDIA Corporation, «About CUDA,» 2022. [En línea]. Available: https://developer.nvidia.com/about-cuda.

[31] S. K. Pathuri, N. Anbazhagan, G. . P. Joshi y J. You, «Feature-Based Sentimental Analysis on Public Attention towards COVID-19 Using CUDA-SADBM Classification Model,» *Sensors,* vol. XII, nº 80, p. 80, 2022.

[32] NVIDIA Corporation, «NVIDIA cuDNN,» 2022. [En línea]. Available: https://developer.nvidia.com/cudnn.

[33] Y. Chen, Y. He, J. Wang, W. Li, L. Xing, X. Zhang y G. Shi, «Automated cone photoreceptor cell identification in confocal adaptive optics scanning laser ophthalmoscope images based on object detection,» *Journal of Innovative Optical Health Sciences,* vol. XV, nº 1, pp. 2250001-1 – 2250001-7, 2022.

[34] S. El Kohli, Y. Jannaj, M. Maanan y H. Rhinane, «DEEP LEARNING: NEW APPROACH FOR DETECTING SCHOLAR EXAMS FRAUD,» *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences,* Vols. %1 de %2XLVI-4-W3-2021, pp. 103-107, 2022.

[35] C. Wang y Z. Wang, «Progressive Multi-Scale Vision Transformer for Facial

Action Unit Detection,» *Frontiers in Neurorobotics,* vol. XV, 2022.

[36] Z. He, Z. Zhang y C. Jung, «Fast Fourier Transform Networks for Object Tracking Based on Correlation Filter,» *IEEE Access,* vol. VI, pp. 6594-6601, 2018.

[37] T. Nishimura, Y. Suzuki, T. Tsuji y T. Watanabe, «Fluid Pressure Monitoring-Based Strategy for Delicate Grasping of Fragile Objects by A Robotic Hand with Fluid Fingertips,» *Sensors,* vol. XIX, nº 4, p. 782, 2019.

[38] G. J. Ramôa, V. Lopes, A. L. Alexandre y S. Mogo, «Real-time 2D–3D door detection and state classification on a low-power device,» *SN Applied Sciences,* vol. III, nº 5, pp. 1-3, 2021.

[39] R. Gandhi, «R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms,» Towards Data Science, 9 Julio 2018. [En línea]. Available: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e.

[40] tzutalin, «LabelImg,» 3 Dciembre 2018. [En línea]. Available: https://github.com/tzutalin/labelImg.

[41] The Jupyter Trademark , «Jupyter.org,» 2022. [En línea]. Available: https://jupyter.org/.

[42] AlexeyAB, «Darknet,» 2021. [En línea]. Available: https://github.com/pjreddie/darknet.

[43] A. Bochkovskiy, C.-Y. Wang y H.-Y. M. Liao, «YOLOv4: Optimal Speed and Accuracy of Object Detection,» *CoRR,* 2020.

[44] Python Software Foundation, «Python,» 2022. [En línea]. Available: https://www.python.org/.

[45] M. D. C. F. M. J. Á. Massiris, «Detección de equipos de protección personal mediante red neuronal convolucional YOLO,» de *Actas de las XXXIX Jornadas de Automática*, 2018.

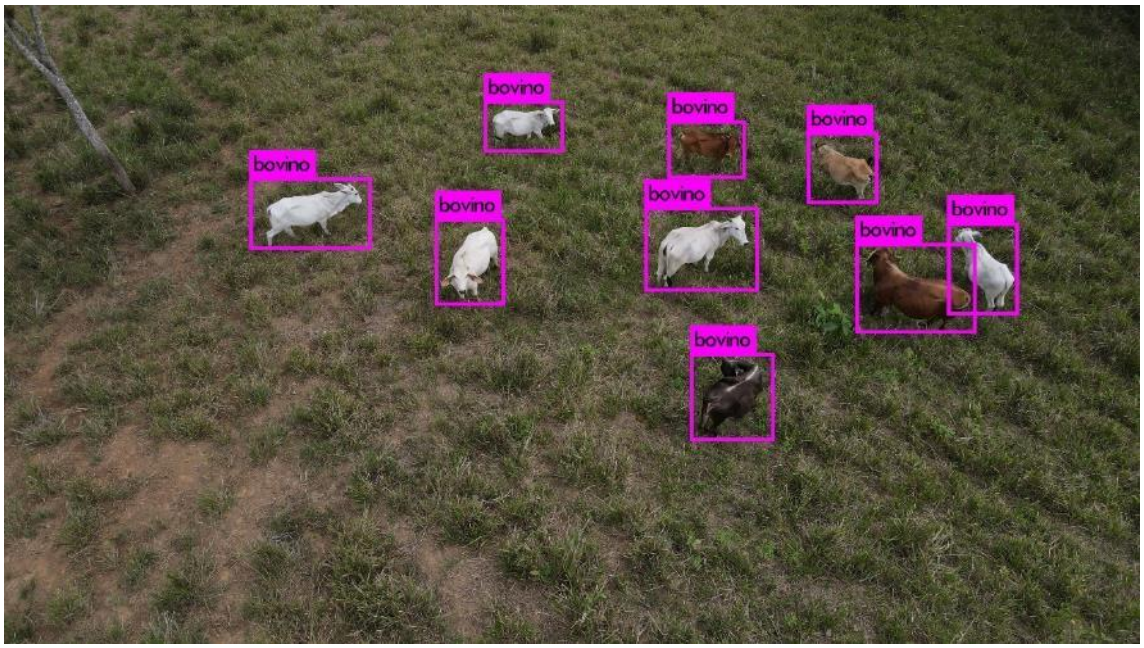
[46] S. M. E. M. S. K. C. J. M. D. W. D. W. J. G. E. L. S. Shima Nofallah, «Machine Learning Techniques for Mitoses Classification,» *Computerized Medical Imaging and Graphics,* vol. LXXXVII, p. 101832, 2021.

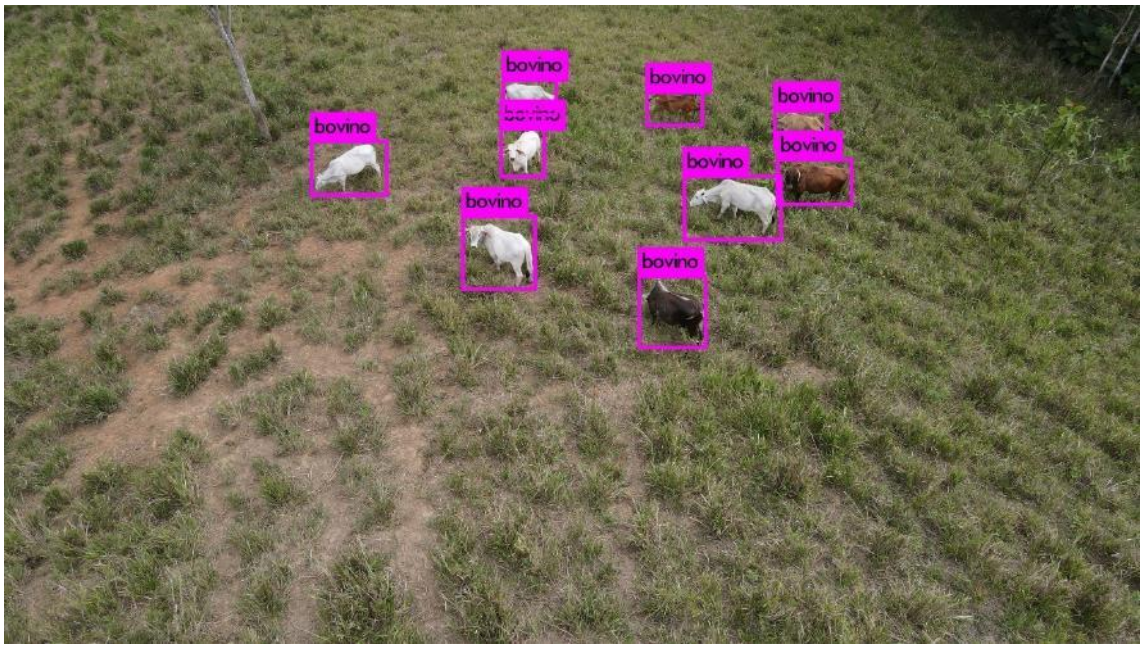
[47] A. M. L. Peña, «YOLO Object Detector for Onboard Driving Images,» 28 Junio 2017. [En línea]. Available: https://ddd.uab.cat/record/181557.

5. ANEXOS

Anexo 1: Detección del modelo obtenido del entrenamiento 0.

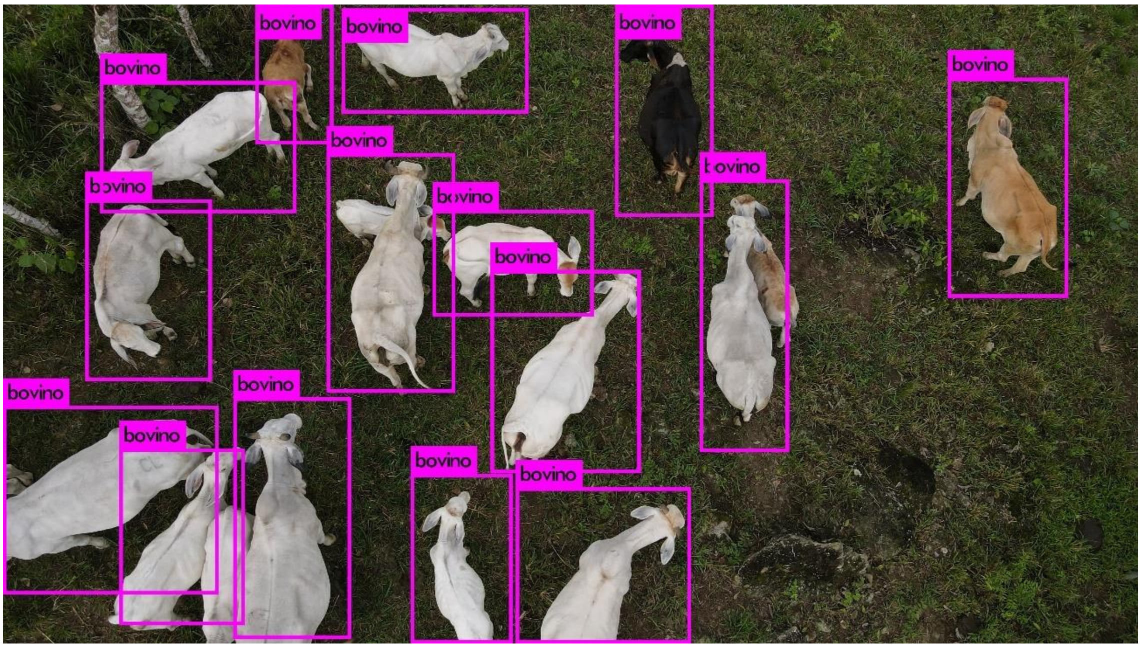
Anexo 2: Detección del modelo obtenido del entrenamiento 1.



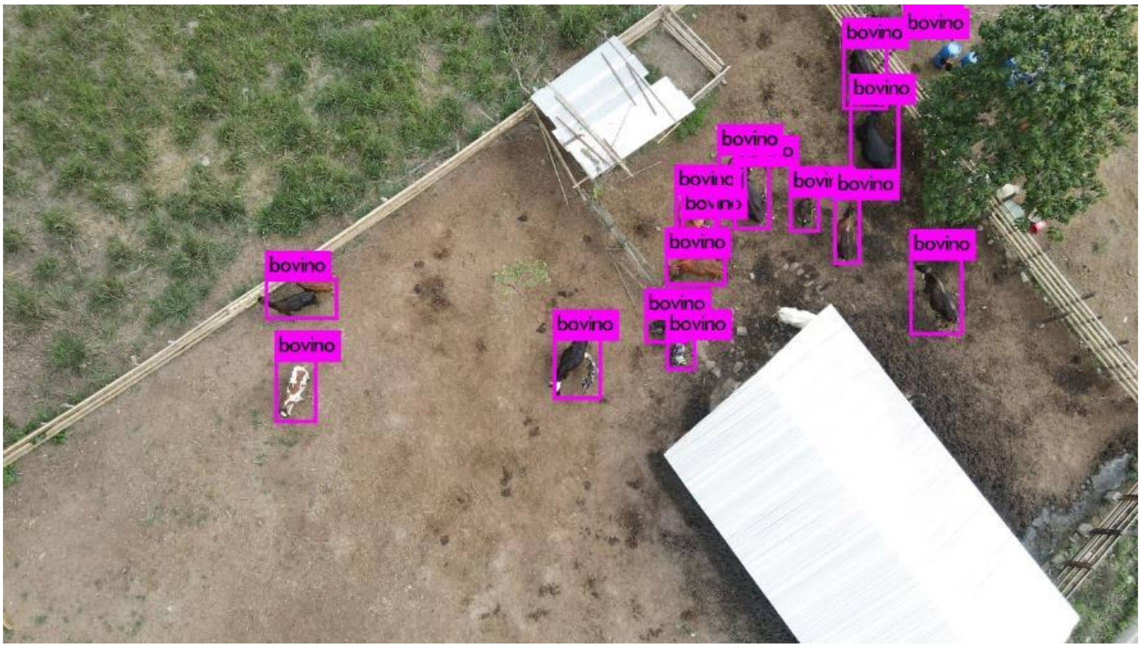


Anexo 3: Detección del modelo obtenido del entrenamiento 2.

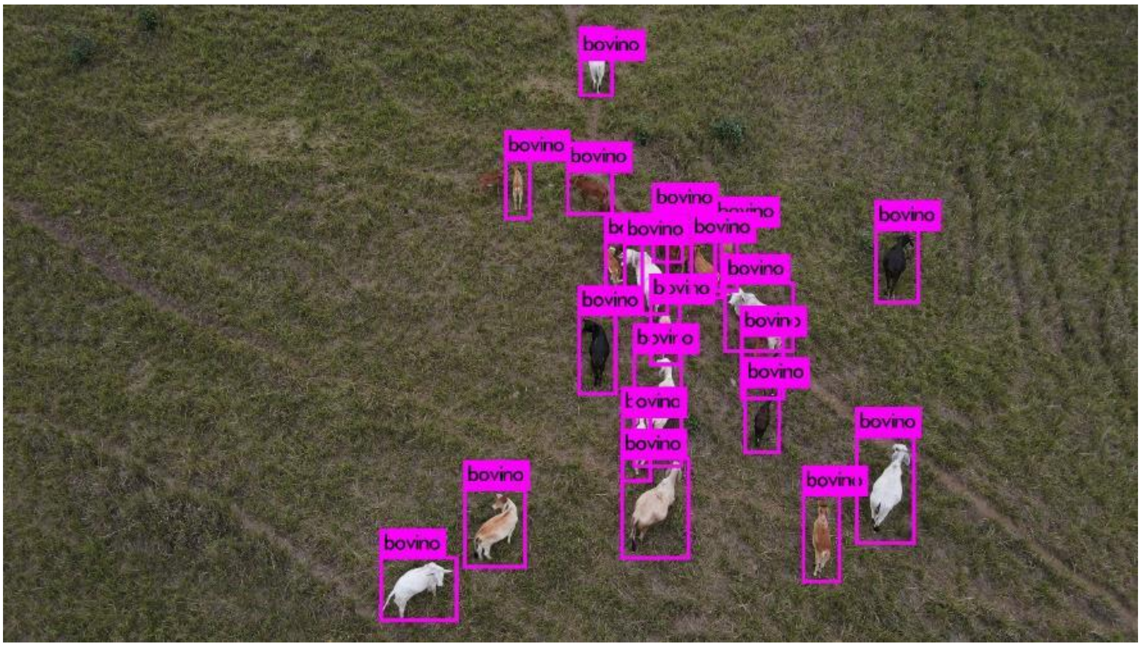
Anexo 4: Detección del modelo obtenido del entrenamiento 3.



Anexo 5: Detección del modelo obtenido del entrenamiento 4.



Anexo 6: Detección del modelo obtenido del entrenamiento 5.



Anexo 7: Detección del modelo obtenido del entrenamiento 6.

Anexo 8: Detección del modelo obtenido del entrenamiento 7.