

Alejandro Aristizábal  
Juan David Rico  
Alexander Hernández  
Juan Felipe Jiménez

## Proyecto – Desarrollo de Soluciones

### Proyecto – Entrega 2

Link a repositorio Github: <https://github.com/jfjimenezc/Proyecto-Desarrollo-de-Soluciones-Grupo6>

[Link a MLFlow](#)

**Contexto del problema:** La conservación de la biodiversidad en África subsahariana enfrenta el reto del conflicto entre fauna silvestre y ganado, agravado por la expansión humana y la falta de monitoreo preciso. Este proyecto busca desarrollar un modelo de aprendizaje profundo para la identificación de especies y el conteo automático de animales en imágenes aéreas. Su impacto será ambiental (monitoreo preciso para conservación), social (reducción de conflictos entre comunidades y fauna) y tecnológico (mejoras en modelos de conteo en entornos complejos).

La validación se realizará con una base de datos pública y métricas como Precisión, Recall y F1-Score, comparando el desempeño con HerdNet. Este avance contribuirá a la gestión sostenible de la biodiversidad y los recursos naturales en la región.

#### Pregunta de negocio y alcance del proyecto:

Teniendo en cuenta el contexto anterior, la pregunta de negocio que guiará nuestro proyecto es: **¿Es posible monitorear las migraciones de manadas de animales, detectando el número de animales y su especie con una precisión aceptable?**

Para definir mejor el término “aceptable” de la pregunta de negocio, nos referimos a lograr igualar las métricas obtenidas por una implementación previa de HerdNet:

F1-Score (%)	MAE	RMSE	AC ( <i>Confusión Promedio</i> ) (%)
83.5	1.9	3.6	7.8

**El alcance del proyecto se mantiene igual que en lo indicado en la entrega 1:**

- Realización de experimentos con los modelos empleando MLFlow para el versionamiento de los modelos y los resultados.
- Desarrollar una maqueta del modelo
- Desarrollar nuevas versiones del modelo, llevando el registro para poder seleccionar las mejores alternativas.
- Empaquetar, integrar y desplegar la última versión del tablero y los modelos.

- Presentación final

### **Conjunto de datos a emplear:**

Mantenemos el mismo conjunto de datos descrito en detalle en la entrega 1. Las imágenes serán tomadas del siguiente [repositorio público](#). Está compuesto por imágenes aéreas de animales de seis especies diferentes, distribuidos en tres subgrupos: Entrenamiento (928 Imágenes), Validación (111) y pruebas (258). Hasta este punto no se contemplan cambios frente a la primera entrega.

### **Modelos a explorar:**

A manera de contexto, investigamos los modelos que han mostrado mejores resultados para detección y clasificación de objetos, encontrando que las redes convolucionales presentan los mejores resultados por lo que profundizamos en el funcionamiento de estos modelos:

**Faster RCNN - Region-based Convolutional Neural Network:** Su funcionamiento se resume en los siguientes pasos:

- **Extracción de características:** Una red convolucional pre-entrenada extrae características de la imagen de entrada, generando un mapa de características.
- **Red de Propuestas de Región (RPN):** a partir del mapa de características, la RPN predice cajas candidatas y sus puntajes de confianza de contener un objeto, reduciendo la cantidad de regiones a procesar.
- **Rol Pooling:** Las regiones propuestas se normalizan a un tamaño fijo.
- **Clasificación y refinamiento:** Un clasificador asigna una categoría a cada región candidata y ajusta sus coordenadas para mejorar la precisión de la caja delimitadora.
- **Salida del modelo:** Se aplican técnicas como Non-Maximum Suppression (NMS) para eliminar regiones redundantes y producir las detecciones finales con sus respectivas clases y coordenadas.

Posteriormente exploramos el modelo de aprendizaje profundo propuesto por HerdNet, el cual utiliza redes neuronales convolucionales para detectar a los animales en la imagen y clasificar su especie.

### **Principales retos:**

Algunos de los principales retos que deben enfrentar este tipo de modelos son oclusiones, dificultades para diferenciar los objetos de interés de los fondos complejos y las distribuciones no uniformes de escala. A continuación profundizamos en cada uno de estos retos:

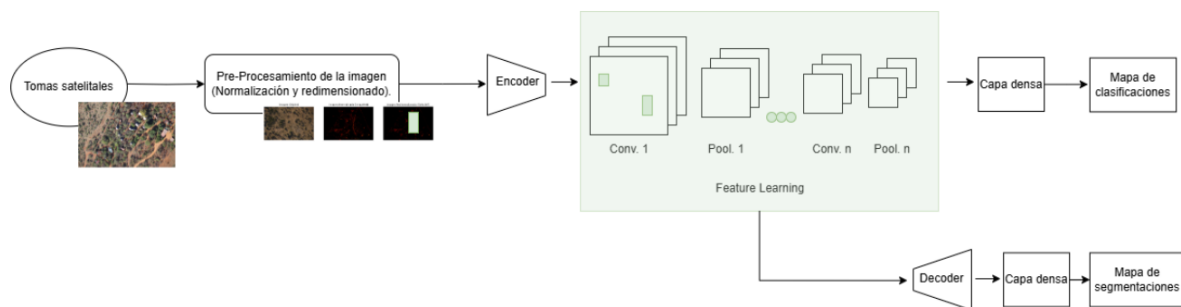
- Oclusiones:** En la medida que la cantidad de animales u otros objetos en la imagen incrementa, los individuos pueden estar cubiertos parcialmente por otros animales u objetos presentes en la imagen dificultando la identificación de la especie y el conteo de los individuos.
- Fondos complejos:** Las imágenes aéreas contienen fondos complejos de interpretar debido a la alta variedad de objetos confusos como sombras, rocas, cuerpos de agua, etc.
- Variación de escala:** El tamaño de los animales varía tanto entre animales de la misma especie como animales de diferentes especies, aumentando la dificultad de detección y clasificación.

- D. **Distribuciones no uniformes:** Debido a que se pueden encontrar diversas distribuciones y niveles de densidad en las manadas en las imágenes y está dificultad se incrementa al tener en cuenta que los dataset cuentan con una gran cantidad de imágenes que contienen pocos animales.

Otra discusión interesante es la de la conveniencia de utilizar puntos en lugar de bounding boxes para marcar los objetos de interés en las imágenes de entrenamiento, encontrando papers como [Kellenberger et al., 2018](#), [Kellenberger et al., 2019b](#), [Kellenberger et al., 2021](#) en el que se respalda el uso de puntos sobre *bounding boxes* debido a que pueden presentar ventajas en imágenes de baja resolución y facilitar el conteo de individuos en grupos de alta densidad.

### Complemento a la maqueta presentada en la entrega 1:

En la primera entrega se propuso la primera versión de nuestra maqueta del prototipo:



### Contemplaba los siguientes elementos:

1. **Entradas:** La entrada del sistema son imágenes aéreas de alta resolución que contienen información del terreno donde se encuentran los animales.
2. **Procesamiento de datos:**
  - A. **Normalización:** Ajusta los valores de píxeles para mejorar el rendimiento del modelo.
  - B. **Redimensionamiento:** Se ajustan las imágenes a un tamaño compatible con la CNN.
3. **Modelamiento:**
  - A. **Encoder:** Mapeo de características a espacio de alta dimensión.
  - B. **CNN:** Luego se aplican capas de convolución y pooling para extraer las características más relevantes de las imágenes, reduciendo la dimensionalidad y manteniendo la información más útil.
4. **Detección de objetos:** Predice cajas delimitadoras de los animales en la imagen.
5. **Decoder (Reconstrucción y Segmentación):** Genera un mapa de segmentación donde cada píxel es clasificado según la especie detectada o la clase 0 que es terreno.

### Modificaciones realizadas:

1. **Detección de objetos:** debido a la información encontrada sobre los modelos y los retos que plantean este tipo de casos de uso, decidimos utilizar el punteo para señalar los

puntos de interés en las imágenes en lugar de las cajas delimitadoras. Lo anterior dado que el etiquetado utilizando bounding boxes es más costoso y se ha encontrado que es menos efectivo en imágenes con altos niveles de oclusión.

2. **Presentación de resultados:** Exploramos ideas de tableros para presentar los resultados del modelo.

### Resultados del modelo:

**Primera inferencia:** Para optimizar los tiempos de la entrega, se realizó la inferencia utilizando únicamente el grupo de datos de validación que contempla 111 imágenes, obteniendo los siguientes resultados:

El modelo identifica los animales y son marcados por medio de puntos:

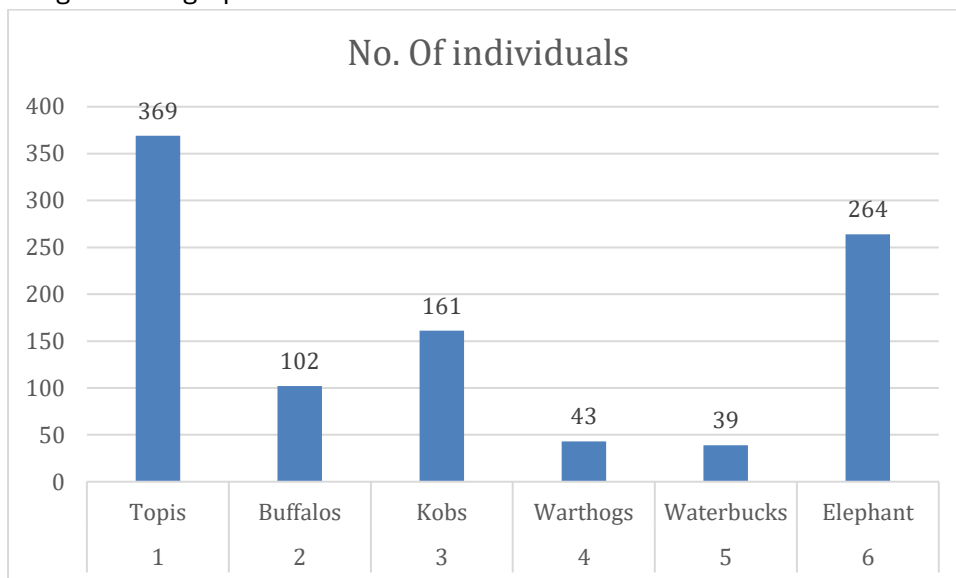


Se clasifican las especies de los animales asignando un porcentaje de certeza:





Se identifica la cantidad de individuos de cada una de las especies que se encuentran en las 111 imágenes del grupo de validación:



Se obtienen los resultados de inferencia de las métricas del primer experimento:

	class	n	recall	precision	f1_score	confusion	mae	mse	rmse	ap
0	1	369	0.897019	0.823383	0.858625	0.048851	1.457143	4.771429	2.184360	0.791720
1	2	102	0.823529	0.792453	0.807692	0.023256	1.111111	2.444444	1.563472	0.724577
2	3	161	0.863354	0.794286	0.827381	0.085526	1.111111	3.444444	1.855921	0.807510
3	4	43	0.418605	0.486486	0.450000	0.000000	1.461538	4.153846	2.038099	0.298429
4	5	39	0.487179	0.826087	0.612903	0.457143	2.200000	7.000000	2.645751	0.437983
5	6	264	0.727273	0.662069	0.693141	0.000000	2.325581	13.860465	3.722965	0.634846
6	binary	978	0.849693	0.804453	0.826455	0.000000	1.738739	8.045045	2.836379	0.702414

## Registro de inferencia en el MLFlow:

The top screenshot shows the MLFlow 'Experiments' view. It displays a table of runs with columns: Run Name, Created, Dataset, Duration, and Source. One run is visible: 'Herdnet\_inference\_from\_colab', created 3 minutes ago, with a duration of 13.1s. The bottom screenshot shows the 'Overview' view for the run 'Herdnet\_inference\_from\_colab'. It includes a description field (currently empty) and a details table.

Details	
Created at	2025-03-02 19:07:15
Created by	root
Experiment ID	0
Status	Finished
Run ID	98dc32df59214bc588ad2ef3abe40fda
Duration	13.1s
Datasets used	—

### Experimento 1:

Se tomó un conjunto de más de 100 imágenes del conjunto de entrenamiento para entrenar el modelo y posteriormente validarlo utilizando el conjunto de validación.

Como paso inicial, se clona el repositorio de HerdNet y se instala el paquete para luego descargar un archivo .zip que contienen parte de las imágenes de experimentación y segmentadas para entrenamiento, evaluación y pruebas.


Como pre-procesamiento de las imágenes, y para facilitar el procesamiento del modelo, se implementó un procedimiento llamado “Patcher”, el cual es una herramienta perteneciente al repositorio de HerdNet que se usa para segmentar las imágenes en más de 80 subdivisiones de cada una. Luego, a los conjuntos de datos se les aplican transformaciones con Albumentations y se convierten en anotaciones. Adicional, se hacen las configuraciones de los dataloaders con la mejor adecuación para la red.

En cuanto a la función de pérdida, se considera un Focal Loss para la cabeza de localización, y un Cross Entropy Loss para la cabeza de clasificación a modo de vectores.



Para la configuración del entrenamiento, se configura el optimizador de Adam con una cierta tasa de aprendizaje y weight decay. También se definen métricas de evaluación para procesar las imágenes, ya hechas patches, o segmentos.

Luego del entrenamiento, se descarga otro archivo de checkpoint para evaluar el modelo y se crea un evaluador para el conjunto de prueba, obteniendo el F1 del score global y las detecciones, los cuales, similar al anterior ejercicio, se obtuvieron los siguientes resultados:



	class	n	recall	precision	f1_score	confusion	mae	mse	rmse	ap
0	1	675	0.875556	0.838298	0.856522	0.057416	1.718310	7.887324	2.808438	0.801938
1	2	349	0.727794	0.916968	0.811502	0.066176	3.500000	35.642857	5.970164	0.693974
2	3	477	0.903564	0.848425	0.875127	0.092632	1.028302	3.952830	1.988173	0.859655
3	4	74	0.445946	0.388235	0.415094	0.057143	2.243243	9.810811	3.132221	0.272273
4	5	36	0.694444	0.657895	0.675676	0.193548	1.230769	2.153846	1.467599	0.595822
5	6	688	0.728198	0.675202	0.700699	0.001992	2.601942	19.669903	4.435076	0.655523
6	binary	2299	0.844715	0.824628	0.834551	0.000000	1.906977	13.542636	3.680032	0.722731

De acuerdo con los resultados anteriores, podemos ver que la implementación del modelo con el entrenamiento aventaja al de la anterior experimentación en el sentido que presenta un F1 Score y Recall ligeramente mejores con poca diferencia, pero hay que tener en cuenta que, al ver el número de caracteres analizados, esta experimentación tuvo un procedimiento más robusto al tener no solo que haber tenido un entrenamiento previo, sino que el ejercicio de clasificación lo hizo con un número mayor al ejercicio de inferencia, por lo que los resultados, a pesar de ser ligeramente más óptimos, representan una significancia en su eficacia dada la variable de caracteres procesados.

Es de anotar que este ejercicio ha hecho que el equipo de trabajo tuviera un reconocimiento al detalle del procesamiento del algoritmo de HerdNet para procesamiento de imágenes y detección de clases.

Así mismo, este ejercicio ha hecho que el equipo considerara las implicaciones significativas al momento de ejecutar este tipo de ejercicios, sobre todo en el cargue computacional, dado a que, a pesar de la presentación de dos experimentaciones generadas, el equipo hizo varias iteraciones para correr el modelo, en donde se pudo evidenciar que la limitante computacional es una situación que no solo ha tenido que afrontar en este ejercicio de acercamiento, más aún para el momento de ejecutar más iteraciones del modelo a corto plazo, que a pesar que estos fueron ejecutados con una máquina de cómputo potente (A100 GPU), se evidenció que en varias ocasiones el sistema colapsaba cuando se querían generar iteraciones del entrenamiento, como adicionar una red neuronal al modelo, o cambiar los inicializadores o los optimizadores.

A pesar de las complicaciones, se logró ver que la mejor manera de preprocesar las imágenes es por medio de la segmentación de estas. Hubo una diferencia muy significativa en el cargue computacional cuando se procesaron segmentadas en contraste al procesamiento de las imágenes enteras, lo cual hacía que el modelo procesara de una manera más pesada. Así mismo, concluimos que el preprocesamiento de las imágenes por medio de la segmentación no perjudica el rendimiento del modelo, sino que lo puede incluso mejorarlo dado a que los patches acercan la imagen a una resolución suficiente y mejora la visualización del animal.

**Reporte de trabajo en equipo:** la exploración y entendimiento de los modelos disponibles, estuvo a cargo de Juan Felipe Jiménez, el ajuste a la maqueta estuvo a cargo de Alejandro Aristizábal, la inferencia inicial estuvo a cargo de Juan David rico y el entrenamiento del modelo inicial estuvo a cargo de Alexander Hernández con aportes de todos los miembros del equipo para la definición de los hiperparametros.

**Bibliografía:**

- Delplanque et al., 2022, Eikelboom et al., 2019, Kellenberger et al., 2017, Kellenberger et al., 2018, Kellenberger et al., 2019a, Naudé and Joubert, 2019, Peng et al., 2020, Torney et al., 2019
- Gao et al., 2020
- Lempitsky and Zisserman (2010)
- Li et al., 2021, Liu et al., 2018