**Alejandro Aristizábal, Juan David Rico, Alexander Hernández, Juan Felipe Jiménez**

**Proyecto – Desarrollo de Soluciones – Grupo 5**

**Proyecto – Entrega 3**

**Link a repositorio Github:** <https://github.com/jfjimenezc/Proyecto-Desarrollo-de-Soluciones-Grupo5.git>

[Link a MLFlow](http://ec2-50-16-4-117.compute-1.amazonaws.com:5000/#/experiments/0?searchFilter=&orderByKey=attributes.start_time&orderByAsc=false&startTime=ALL&lifecycleFilter=Active&modelVersionFilter=All+Runs&datasetsFilter=W10%3D)

**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 (Confusión Promedio) (%) |
| 83.5 | 1.9 | 3.6 | 7.8 |

**El alcance del proyecto se mantiene igual que en lo indicado en las entregas anteriores:**

* + 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](https://dataverse.uliege.be/file.xhtml?fileId=11098&version=1.0). 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 explorados:**

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 imágen 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:

1. **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.
2. **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.
3. **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.
4. **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](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub#b0100), [Kellenberger et al., 2019b](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0110), [Kellenberger et al., 2021](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0115) 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 2:**

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

A diagram of a diagram

AI-generated content may be incorrect.**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:**
   1. **Normalización:** Ajusta los valores de píxeles para mejorar el rendimiento del modelo.
   2. **Redimensionamiento:** Se ajustan las imágenes a un tamaño compatible con la CNN.
3. **Modelamiento:**
   1. **Encoder:** Mapeo de características a espacio de alta dimensión.
   2. **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:** 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 cajas delimitadoras es más costoso y se ha encontrado que es menos efectivo en imágenes con altos niveles de oclusión.
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.
6. **Creación del tablero para facilitar el uso del modelo:** Ver manual de usuario. Se plantea un mock up del tablero que se seguirá perfeccionando en el próximo ciclo en la medida en que sigamos avanzando en el proyecto de la maestría.

**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:



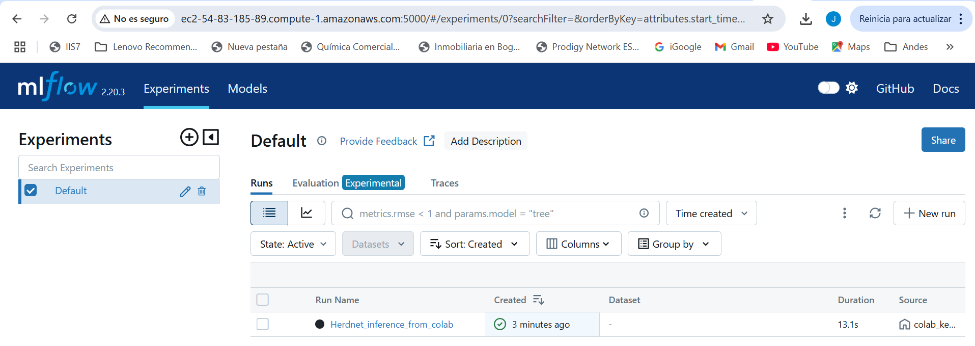
Se clasifican las especies de los animales asignando un porcentaje de certeza:

An aerial view of a green field

AI-generated content may be incorrect. An elephant in the forest

AI-generated content may be incorrect.

**Registro de inferencia en el MLFlow:**



A screenshot of a computer

AI-generated content may be incorrect.

**Experimentaciones[[1]](#footnote-2):**

* **Preparación de la data:**

Inicialmente contamos con las siguientes cantidades de imágenes por cada dataset: 157 imágenes para entrenamiento, 121 imágenes para evaluación y 111 imágenes para validación, los cuales, pasaron por el siguiente preprocesamiento antes del entrenamiento:

* *Subdivisión de patches:*

Para la preparación de la data y evitar los inconvenientes que presenciamos en la entrega anterior, generamos *patches* en los datasets de entrenamiento, evaluación y validación, esto con el fin de que las segmentaciones de las fotos tuvieran mayor detalle para la máquina al mejorar la cercanía de la imagen sin tener problemas de procesamiento de cómputo. Las imágenes originales se dividieron en patches de 512X512 píxeles, lo cual, una vez generado este proceso de segmentación, las imágenes se multiplicaron en las siguientes cantidades: 748 imágenes para entrenamiento, 600 imágenes de evaluación y 569 imágenes de validación.

* *Conversión de anotaciones:*

Los archivos de anotaciones (.csv) que contienen las ubicaciones de los animales por cada foto fueron transformados para representar cada objeto como un punto central en lugar de bounding boxes (como había sido recibida inicialmente la data). De esta manera, se calcularon las coordenadas (x,y) de cada objeto como el punto medio entre las coordenadas máximas y mínimas que punteaban a la ubicación de cada animal.

* *Aplicación de transformaciones de datos (Data Augmentation):*

En este proceso, se aplicaron técnicas de aumento de datos en el conjunto de entrenamiento usando la librería *albumentations*, lo cual incluyó generar un flip vertical y horizontal con 50% de probabilidad, rotaciones aleatorias de 90 grados, ajustes de brillo y contraste con un 20% de probabilidad, desenfoque aleatorio y normalización de valores de píxeles. En el caso de los sets de validación y prueba, solo se aplicó normalización de las imágenes.

* *Conversión de datos para el modelo:*

Posteriormente, se implementaron anotaciones específicas para la detección de objetos de HerdNet como convertir coordenadas de puntos en mapas de características para el modelo, creación de máscaras binarias a partir de las coordenadas y reducción de la resolución de las imágenes en la validación y prueba para optimización de la memoria y procesamiento.

* *Creación de datasets y dataloaders:*

Por último, se crearon los tres conjuntos de datos y se definieron los Dataloaders para cada conjunto. En el caso de la data de entrenamiento, se cargaron los datos en batches de 2 imágenes con shuffle activado, mientras que en los datasets de evaluación y validación se procesaron una imagen de batch sin mezclar los datos.

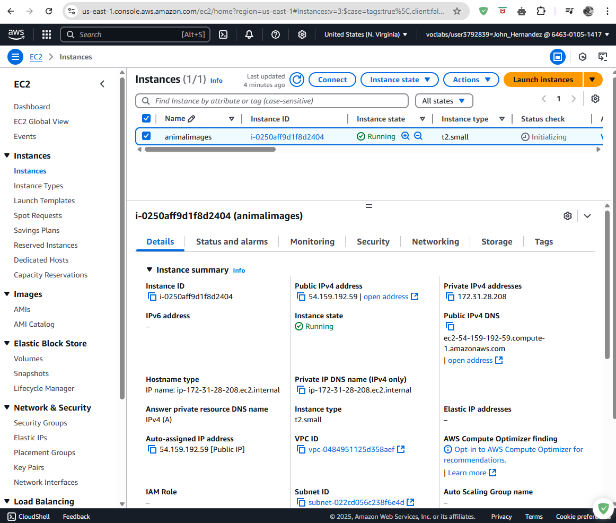
* Una vez tenidas las imágenes segmentadas y procesadas, continuamos con los entrenamientos de las experimentaciones:
* **Entrenamiento modelo - Experimento 0:**

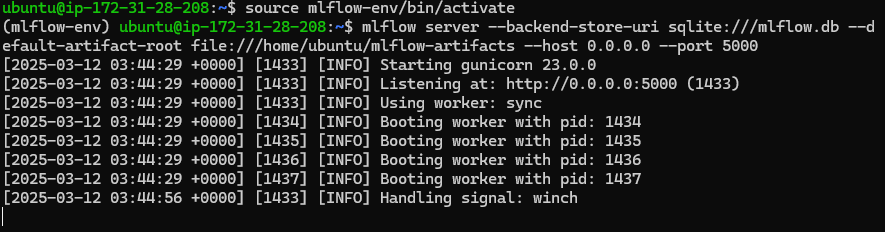
La primera experimentación constó en la inicialización del modelo con los pesos preentrenados con dos configuraciones de pérdida: Uno de pérdida focal para manejo del desbalance en la detección de clases minoritarias, y un Cross Entropy con pesos ajustados y asignados a cada clase para mejorar el aprendizaje. Adicionalmente, se definió un optimizador Adam con una tasa de aprendizaje de 1e-4 y un weight\_decay de 1e-3 para prevenir el sobreajuste.

Para el procesamiento, se definieron 5 épocas de entrenamiento y se configuró el entrenador con su evaluador para cálculo de las métricas de rendimiento, siendo el f1 score la métrica principal.

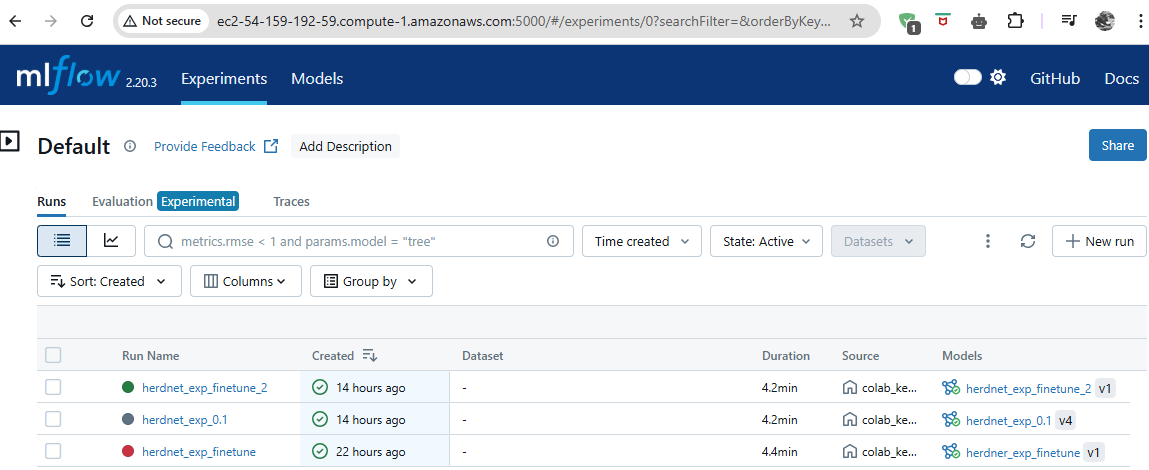
Una vez entrenado el modelo, se cargaron los mejores pesos entenados y se evaluó el modelo en el conjunto de prueba, generando el respectivo archivo .pkl del modelo entrenado y luego se configuró la integración con MLFlow para registro del modelo junto con sus métricas.

A continuación, se muestra la máquina virtual e instancia que sirvieron como conexión con MLFlow:

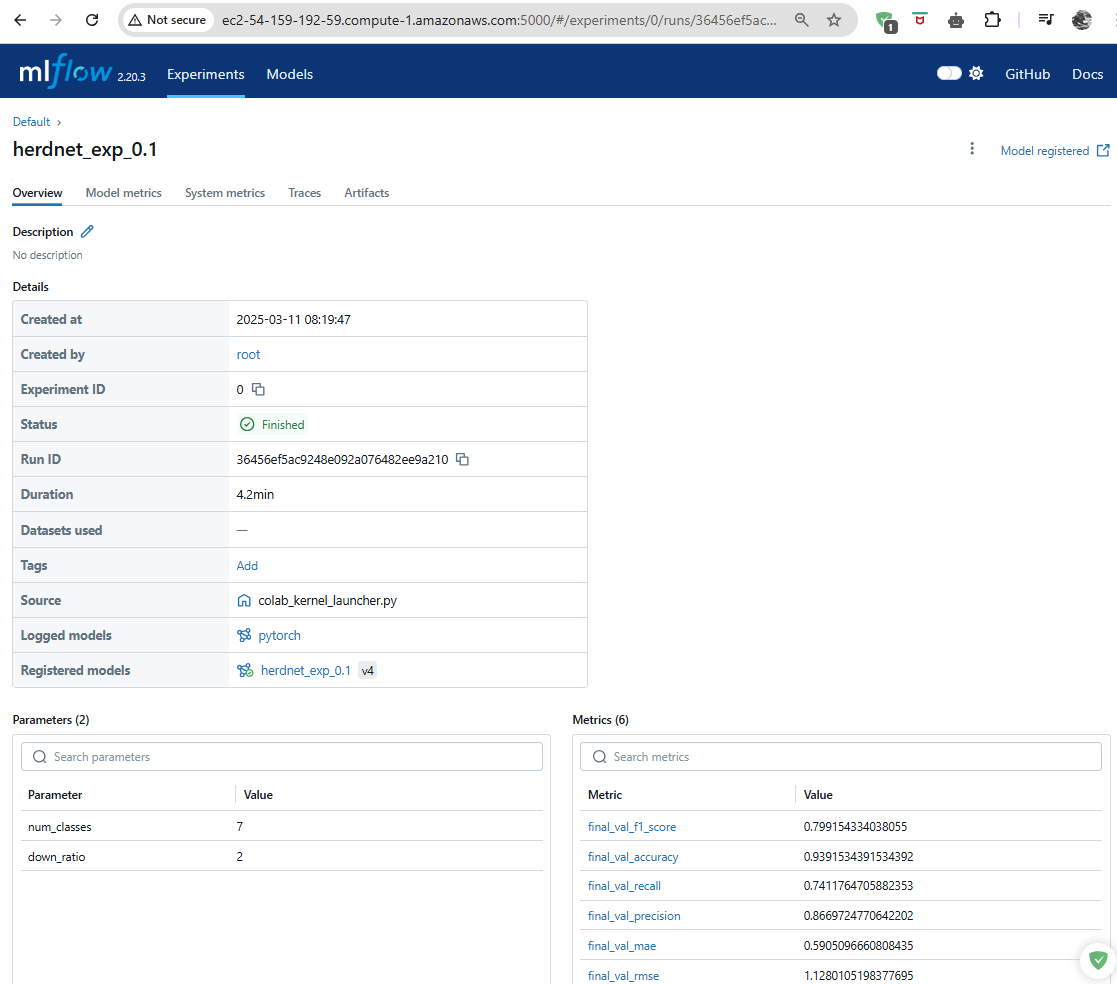




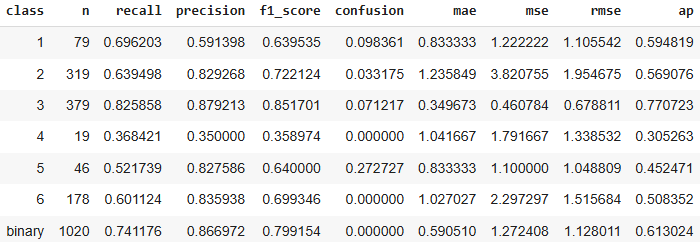
Modelos almacenados en MLFlow:



Y a continuación, los resultados de la primera experimentación registrada en MLFlow:



De acuerdo con los resultados de esta primera experimentación, podemos ver que hay un buen equilibrio entre presición y recall, el cual se ve reflejado en un F1 Score de casi el 80%. Una presición del 86% indica que el modelo es eficiente al identificar las clases, sin embargo, al ver el Recall, vemos que el rendimiento es aceptable, pero no tan eficiente como en la presición, lo que indica que el modelo no está detectando los vasos positivos, por lo que podría estar perdiendo detecciones. Adicionalmente, podemos ver que los índices de MAE y RMSE indican que los errores en la ubicación de las detecciones son bajos en promedio.

Para mayor detalle de los resultados, y enfocándolos por clase, tenemos lo siguiente:

Según los resultados por clase, vemos que la clase 3 tiene un alto desempeño en el F1 Score, precisión y recall, lo que indica que esta clase está bien balanceada en términos de predicción, mientras que la clase 4 tuvo el peor desempelo en las tres métricas, esto se puede deber a un desequilibrio de los datos, ya que el número proporcionado en las imagenes de esta clase fueron los de menor cantidad, lo que dificulta al modelo en diferenciar esta clase. La clase 6 también tiene un recall relativamente bajo aunque su precisión es buena, lo que significa que el modelo rara vez predice esta clase, aunque es acertado.

* **Entrenamiento modelo: Experimento 1:**

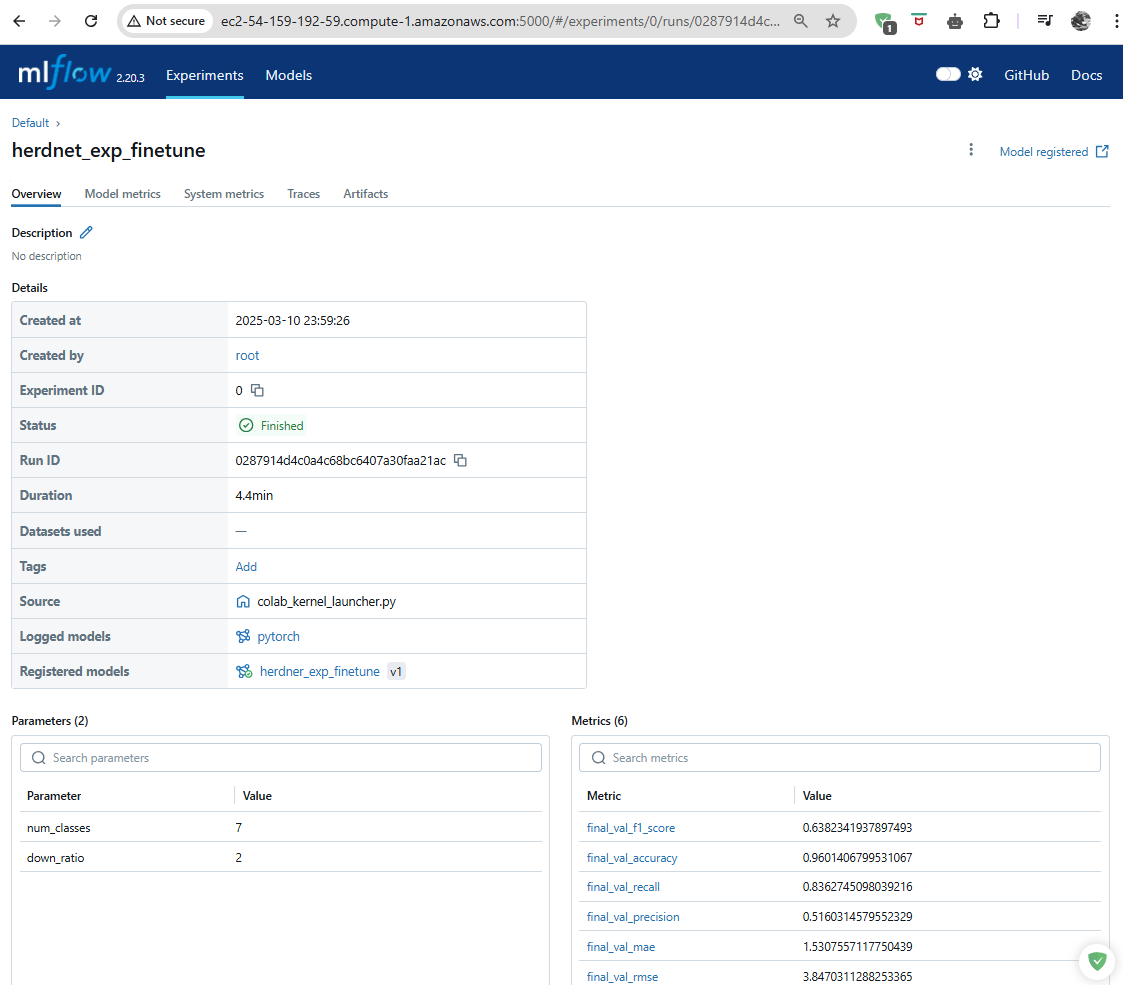
En este segundo modelo se buscó hacer un Fine-Tuning en base al modelo anterior manejando el mismo preprocesamiento de los datos por medio de la optimización únicamente de ciertas partes de la red.

Para este experimento se hicieron cambios clave en el Fine Tuning. Inicialmente, se congelaron todas las capas del modelo para evitar que el entrenamiento afecte los pesos de la parte más profunda de la red y solo se permitieron actualizaciones en las últimas capas del modelo, específicamente en las últimas capas convolucionales del backbone y en capa encargada de la clasificación final.

Adicionalmente, se hizo una reducción de la Tasa de Aprendizaje a 1e-5, más bajo que el modelo anterior con el fin de evitar sobreajuste en las capas superiores, y se redujo la penalización weight decay de 5e-4 para mejorar la estabilidad del entrenamiento.

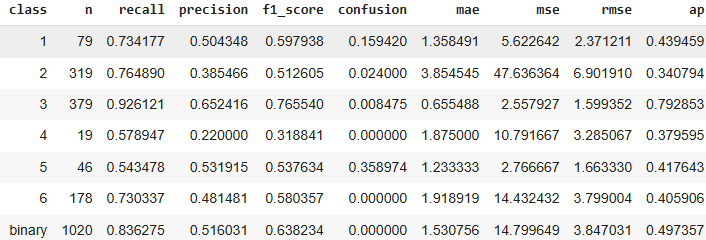
En cuanto al optimizador, se creó un optimizador Adam que afectaría únicamente las capas descongeladas, asegurando que el modelo no pierda el conocimiento previo y se mantuvo el mismo número de épocas.

Resultados experimento1:



De acuerdo con los resultados generales, vemos que el F1 Score global bajó un 16% con respecto al anterior modelo, aunque hubo un aumento de la misma proporción en el Accuracy al igual que el recall, lo que significa que el modelo detectó más positivos correctamente. Sin embargo, la precisión bajó drásticamente, lo que indica que el modelo cometió más falsos positivos. Esto generó que el F1 Score cayera significatigamente, afectando el balance entre recall y precisión. Adicionalmente, el Error Absoluto y el Error Cuadrático Medio aumentaron, lo que indica que hay mayor dispersión y peores predicciones en términos de exactitud.

Resultados experimento 1 por clase:



De acuerdo con los resultados por clase, vemos un aumento en el recall, pero al mismo tiempo una caída en la precisión para las clases 1 y 2. La clase 6 resultó con un RMSE más alto, indicando que el modelo está haciendo predicciones menos precisas en esta clase.

La clase 4 resultó con el peor F1 Score, afectando el rendimiento global, mientras que la Clase 3 mostró el mejor desempeño en el F1 Score, lo que nos indicaría que existe una relación con la cantidad de unidades por clase y el desempeño de predicción.

Este efecto pudo haber ocurrido a que congelar el backbone y solo ajustar las capas superiores afectó la capacidad de generación mientras que la optimización pudo haber favorecido recall sobre precisión, causando una alta tasa de falsos positivos.

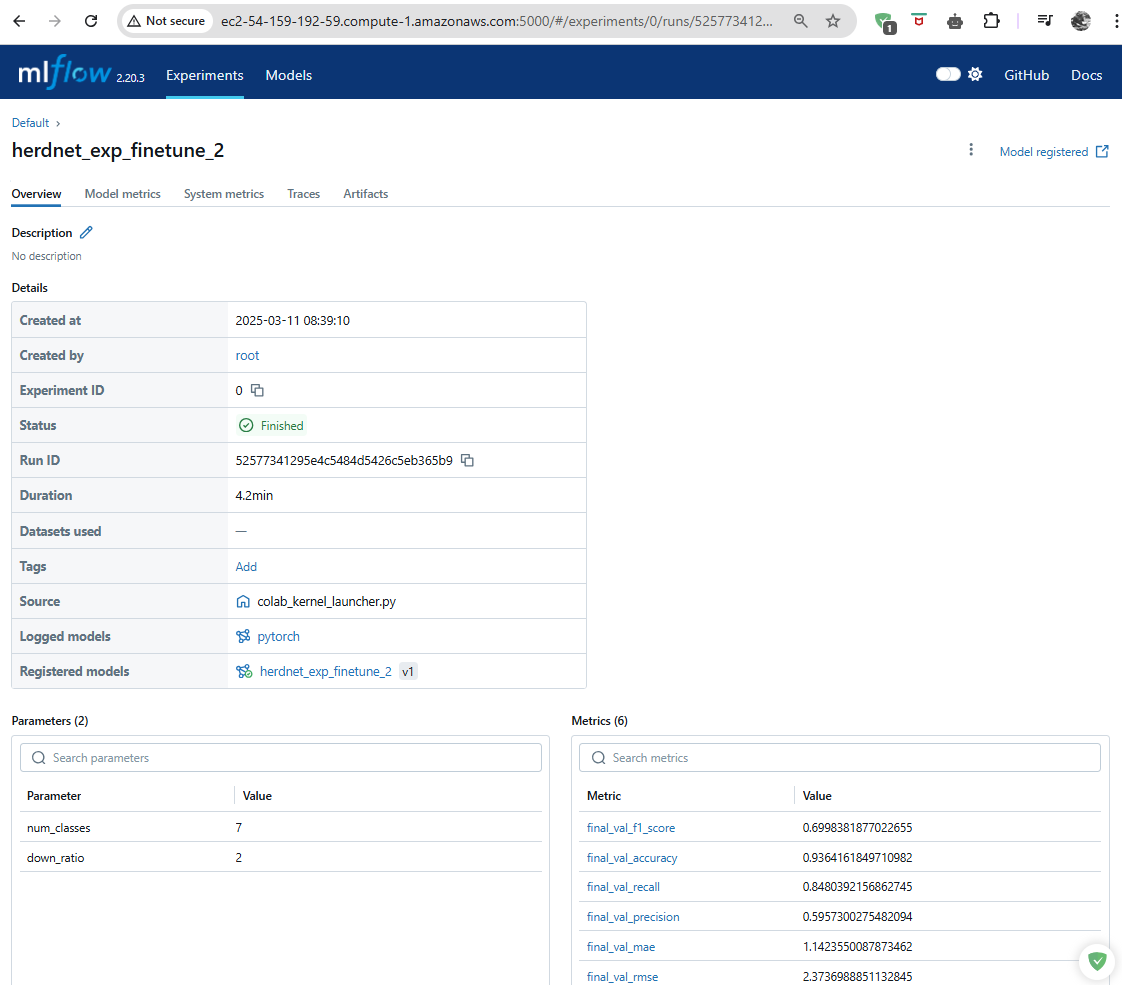
Pudimos ver con este ejercicio que, aunque el modelo mejoró en el recall, sacrificó demasiado en la precisión, lo que afectó el balance general. En otras palabras, aunque el modelo mejoró en la detección de positivos, el alto costo de precisión afectó el rendimiento general. El aumento de las métricas relacionadas con el error sugiere que las predicciones se volvieron un poco más dispersas, lo que sugiere que el modelo tiene una alta dependencia con la estructura del backbone.

* **Entrenamiento modelo: Experimento 2:**

En esta tercera experimentación se congelaron todas las capas del modelo para preservar los pesos previamente entrenados y se descongelaron las capas superiores y la capa clasificatoria para permitir su ajuste en el fine-tuning. Se mantuvo una Tasa de Aprendizaje baja (1e-5) para evitar sobreajuste y se fijó un Weight Decay igualmente bajo (5e-4) para regularizar el modelo.

En esta ocasión se incrementaron el número de épocas (10) al doble de las anteriores experimentaciones y se utilizó el optimizador AdamW con estas configuraciones.

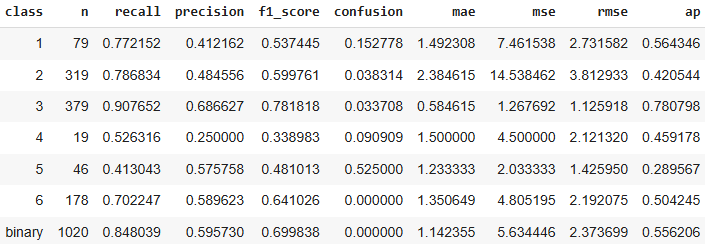
Algo particular que se hizo en este experimento fue aplicar tasas de aprendizaje diferenciadas para distintas partes del modelo: 5e-6 para el cuarto nivel del backbone, 1e-5 para el quinto nivel del backbone y 5e-5 para la capa final clasificatoria. Luego se implementa un scheduler que ajusta la tasa de aprendizaje si el modelo deja de mejorar en la métrica de validación.



De acuerdo con los resultados globales de este experimento, vemos que el F1 Score general fue mejor que el del primer fine-tuning. El recall aumentó igualmente, aunque la precisión, a pesar que tuvo mejor rendimiento que el primer fine tuning, se mantuvo bajo, lo que sugiere que el modelo aun genera falsos positivos.

En cuanto a las métricas relacionadas con el error, vemos que estos disminuyeron, lo que indica que las predicciones del modelo se acercan más a las etiquetas reales y que hay una reducción en la dispersión del error.

Evaluación por clase:



Ahora vemos que las clases 3 y 6 siguen mostrando los mejores F1 Scores, mientras que las clases 1 y 2 mejoraron su recall y precision con respecto al anterior fine tuning. La clase 4 sigue siendo la de menor balance.

En este segundo experimento de fine tuning vemos que las métricas mejoraron, principalmente en F1 Score y MAE, aunque la precisión sigue siendo un área que se puede mejorar. Esto puede indicar que el modelo está generando más detecciones correctas, pero aún tiene margen de optimización en la reducción de falsos positivos. Adicionalmente, vimos que más épocas de iteración generan también mejores resultados.

**Conclusiones generales de las experimentaciones:**

De acuerdo con las experimentaciones y sus resultados, podemos decir que el primer modelo de experimentación presentó un buen desempeño general en un subconjunto de fotos seleccionado por el equipo de trabajo, aunque obtuvo un recall relativamente bajo, lo que significa que no detecta algunas instancias correctamente. Sin embargo, su rendimiento general fue bueno. El primer fine tuning mejoró el recall, pero a costa de una caída importante en la precisión, lo que indicó que se detectan más objetos pero a costa de muchos falsos positivos, mientras que en el segundo fine tuning ocurrió que hubo un mejor equilibrio del recall y precisión, esto debido a que se configuraron más capas ajustadas, aumento de épocas y configuración de tasas de aprendizaje diferidas por conjunto de capas junto con un scheduler para ajustar dinámicamente el LR.

Como conclusión, podríamos decir que el modelo del segundo fine tuning tiene una mejor opción para un balance entre recall y precisión, por lo que se podría explorar más ajustes en las tasas de aprendizaje manteniendo su dinámica de configurarlas de manera diferida y progresiva en sus conjuntos de capas y aumentando las épocas para seguir optimizando la precisión sin perder recall.

**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 elaboración del tablero 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, la documentación estuvo a cargo de todos los miembros, de acuerdo con la distribución de las responsabilidades.

**Bibliografía:**

* [Delplanque et al., 2022](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0045), [Eikelboom et al., 2019](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0050), [Kellenberger et al., 2017](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0095), [Kellenberger et al., 2018](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0100), [Kellenberger et al., 2019a](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0105), [Naudé and Joubert, 2019](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0185), [Peng et al., 2020](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0225), [Torney et al., 2019](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0275)

* [Gao et al., 2020](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0340)
* [Lempitsky and Zisserman (2010)](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0140)

* [Li et al., 2021](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0145), [Liu et al., 2018](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub" \l "b0175)
* <https://www.uniandes.edu.co/es/noticias/ingenieria/inteligencia-artificial-para-la-proteccion-del-amazonas>

**A continuación se presentan los demás entregables, unidos a este mismo documento para poder cargarlo en el único campo disponible en Coursera**

**Manual de usuario del tablero**

El objetivo de este documento es ofrecer un conjunto de instrucciones para el uso básico del tablero **Aerial Animal Detector.**

Esta herramienta tiene como propósito procesar imágenes para identificar la especie de los animales presentes en ella y contar la cantidad de individuos presentes en la imagen. El modelo actual ha sido entrenado para las siguientes especies: Topis, Buffalos, Kobs, Wathogs, Waterbucks, Elephant.

**Instrucciones de uso:**

**Ingrese al siguiente enlace:** [**https://aerialanimals.streamlit.app/**](https://aerialanimals.streamlit.app/)

1. **Encontrará las siguientes opciones:**

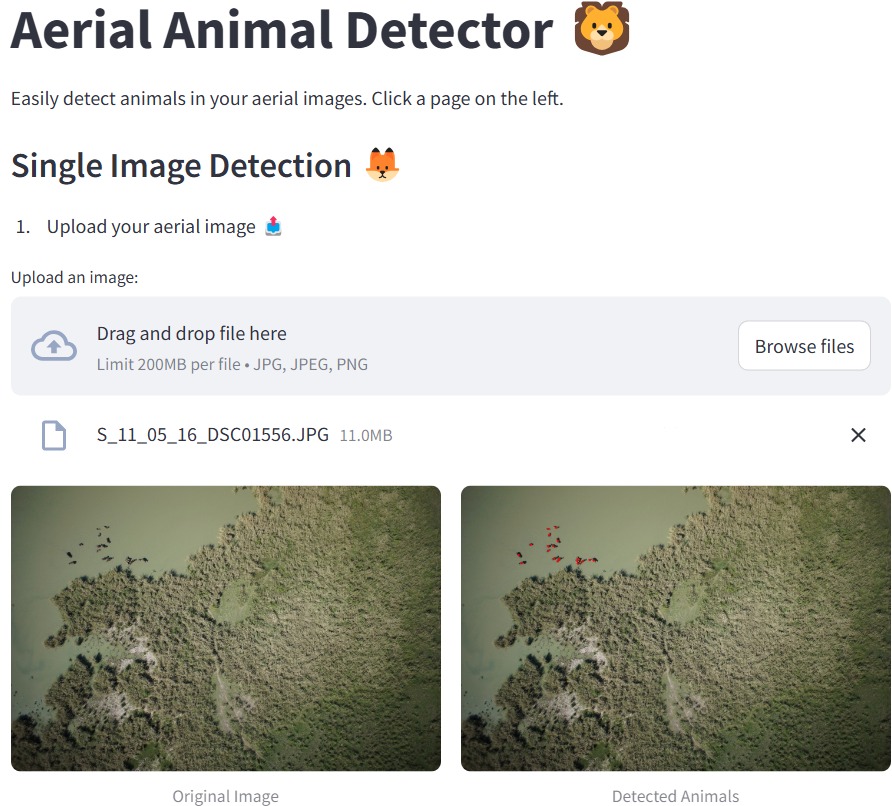
**A screenshot of a computer

AI-generated content may be incorrect.**

**“Single image”:** Cargará una única imagen para ser procesada

**“Batch Prediction”:** Cargará más de una imagen para ser procesadas en simultáneo.

1. **Obtención de resultados:**



**Manual de instalación del tablero:** dado que el tablero se encuentra disponible en el link mencionado en el manual de usuario, no se requieren instrucciones de instalación.

**Link a la presentación final:** [**Presentación Final.pptx**](https://nextant-my.sharepoint.com/:p:/p/jfjimenez/EbIPnhLl8dFHkZcYV60MtAUB4cV8eVqlq4CFENt2LWBPCQ?e=ePzDss)

1. Las experimentaciones 0, los Fine Tuning 1 y 2 se encuentran en los siguientes repositorios: [Experimento 0 y Fine Tuning 1 (Repo)](https://github.com/jfjimenezc/Proyecto-Desarrollo-de-Soluciones-Grupo5/blob/main/Experimento_%26_Fine_Tuning_1.ipynb) y [Fine Tuning 2 (Repo)](https://github.com/jfjimenezc/Proyecto-Desarrollo-de-Soluciones-Grupo5/blob/main/Experimento_0_y_Fine_Tuning_2.ipynb) [↑](#footnote-ref-2)