**Aerial Animal Detector – Identificación de Especies y Conteo de Animales**

**Participantes:**

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

**Introducción:**

**Objetivo general:** Desarrollar un modelo de aprendizaje profundo para la identificación de especies y el conteo automático de animales en imágenes aéreas, con impactos ambientales, sociales y tecnológicos.

**Problema que se aborda:** En el África subsahariana los conflictos entre la fauna silvestre y el ganado han aumentado en las últimas décadas debido a la expansión de la población humana y la creciente demanda de tierras para la ganadería. Estos conflictos generan una competencia directa por recursos como pastizales y fuentes de agua, afectando tanto a la vida silvestre como a los medios de subsistencia de las comunidades rurales. La falta de un monitoreo preciso y continuo de la densidad de animales en estas regiones impide tomar decisiones informadas para equilibrar las necesidades de conservación y desarrollo económico local.

**Solución planteada:** diseñar y entrenar un modelo capaz de detectar y contar con precisión animales de manadas densas, superando las limitaciones de los métodos tradicionales. Utilizaremos una arquitectura especializada en tareas de conteo, optimizando la capacidad del modelo para manejar los desafíos inherentes al problema, como la oclusión parcial de los animales y la distribución no uniforme de los mismos en las imágenes.

**Descripción de la propuesta de solución:**

Dadas las restricciones de tiempo para la ejecución del proyecto y de acuerdo con nuestra investigación realizada en la etapa de preparación, nuestra propuesta se basa en utilizar un modelo ya existente como HerdNet y por medio de técnicas de fine-tunning, buscaremos alcanzar y superar las métricas obtenidas por este modelo:

|  |  |  |  |
| --- | --- | --- | --- |
| F1-Score (%) | MAE | RMSE | AC (*Confusión Promedio*) (%) |
| 83.5 | 1.9 | 3.6 | 7.8 |

Para explicar un poco mejor nuestra propuesta de solución, a continuación presentamos nuestra metodología de desarrollo que sigue un enfoque iterativo basado en experimentación y mejora continua:

* + - 1. **Recolección y Preprocesamiento de Datos:**
* Se descarga el dataset de imágenes público.
* Se aplicarán transformaciones y segmentación de imágenes.
* Conversión de anotaciones de bounding boxes a puntos de referencia.
  + - 1. **Entrenamiento del Modelo:**
* Ajuste inicial del modelo **HerdNet**, manteniendo la arquitectura base.
* Optimización con Adam y ajuste de la tasa de aprendizaje.

**C. Validación y Evaluación:**

* Medición de desempeño en el conjunto de validación mediante métricas F1-Score, Precisión y Recall.
* Comparación con los resultados de HerdNet para evaluar mejoras.

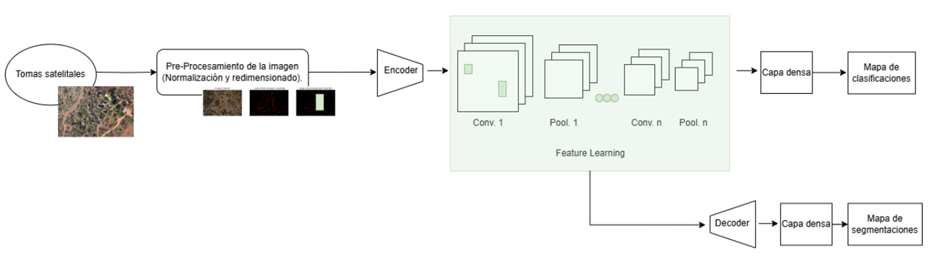
**D. Ajuste y Experimentación:**

* Implementaremos diferentes estrategias de fine-tuning.
* Analizaremos el impacto de la segmentación de imágenes en el rendimiento del modelo.
* Refinaremos los hiperparámetros para mejorar la capacidad de generalización del modelo.

**E. Implementación y Despliegue:**

* Integraremos el modelo con una interfaz de usuario para la carga y análisis de imágenes aéreas.
* Utilizaremos MLFlow y Github para almacenar las mejores versiones del modelo y facilitar su reproducción

A continuación presentamos nuestra solución de manera gráfica para una mejor comprensión:



**Recolección y preparación de datos:**

Comenzamos con las siguientes cantidades de imágenes en cada conjunto de datos: 157 para entrenamiento, 121 para evaluación y 111 para validación. Estas imágenes fueron sometidas a un preprocesamiento previo al entrenamiento del modelo:

* *Generación y subdivisión de patches:*

Para mejorar la calidad de los datos imputados al modelo, dividimos las imágenes del entrenamiento, evaluación y validación en patches (subdivisiones de las imágenes). Este procedimiento se realizó para proporcionar a la máquina segmentaciones más detalladas, permitiendo un acercamiento a la imagen sin complicaciones en el procesamiento. Las imágenes se fraccionaron en patches de 512x512 píxeles, lo que resultó en un aumento en la cantidad de imágenes: 748 para entrenamiento, 600 para evaluación y 569 para validación.

* *Conversión de anotaciones:*

Los archivos de anotaciones (.csv) que contenían las posiciones de los animales en cada imagen fueron transformadas para representar cada objeto como un punto central en lugar de cajas delimitadoras (bounding boxes). Así, se calcularon las coordenadas x,y de cada objeto como el punto medio entre las máximas y mínimas que definían la ubicación de cada animal.

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

Se aplicaron técnicas de aumento de datos al conjunto de entrenamiento utilizando la biblioteca albumentations. Esto incluyó voltear las imágenes vertical y horizontalmente con un 50% de probabilidad, realizar rotaciones aleatorias de 90 grados y ajustar brillo y contraste, además de aplicar desenfoque aleatorio y normalización de los valores de píxeles. Para los conjuntos de validación y prueba, solo se realizó la normalización de las imágenes.

* *Conversión de datos para el modelo:*

Se generaron anotaciones específicas para la detección de objetos mediante HerdNet, que incluyeron transformar las coordenadas de puntos en mapas de características, crear máscaras binarias a partir de estas coordenadas y reducir la resolución de las imágenes en validación y prueba para optimizar el uso de memoria y procesamiento.

* *Creación de datasets y dataloaders:*

Finalmente, se establecieron los tres conjuntos de datos y se definieron los Dataloaders correspondientes. Para los datos de entrenamiento se cargaron en batches de 2 imágenes con activación de shuffle, mientras que en los conjuntos de evaluación y validación se procesó una imagen por batch sin mezclar los datos. Con las imágenes segmentadas y procesadas, procedimos a realizar los entrenamientos correspondientes en nuestras experimentaciones.

Una vez tenidas las imágenes segmentadas y procesadas, continuamos con los entrenamientos de las experimentaciones, las cuales, se han generado de la siguiente manera inicialmente:

* *Experimento 1:*

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 entrenados 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.

* *Experimento 2:*

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.

* *Experimento 3:*

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.

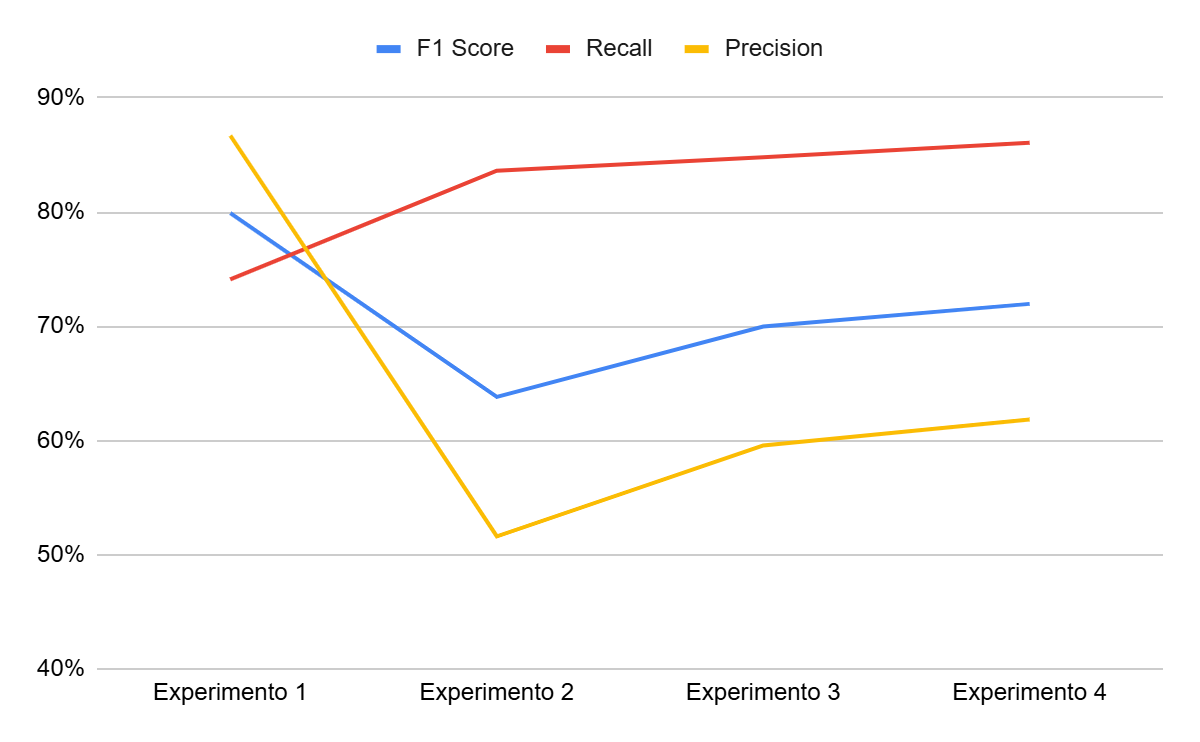
* *Experimento 4:*

En esta cuarta experimentación se trabajó sobre un subconjunto pequeño de 101 imágenes (sub\_train) para explorar hasta qué punto el modelo pre-entrenado podía adaptarse a datos muy específicos.

En este experimento se incrementó el número de épocas a 10 (el doble con respecto a los anteriores experimentos) y se usó el optimizador AdamW con grupos de parámetros y tasas de aprendizaje diferenciadas: 5e-6 para el nivel 4, 1e-5 para el nivel 5, 5e-5 para la capa final y 2e-4 para el weight decay. Se implementó el scheduler ReduceLROnPlateau para reducir la tasa de aprendizaje si el F1 no mejoraba.

**Resultados obtenidos**

En la siguiente gráfica podemos apreciar el rendimiento de los distintos experimentos que se han generado:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Métricas | **Experimento 1** | **Experimento 2** | **Experimento 3** | **Experimento 4** |
| **F1 Score** | 79,91% | 63,82% | 69,98% | 71,96% |
| **Recall** | 74,11% | 83,62% | 84,80% | 86,07% |
| **Precisión** | 86,69% | 51,60% | 59,57% | 61,83% |
| **MAE** | 0,5905 | 1,5307 | 1,1423 | 1.0544 |
| **RMSE** | 1,1280 | 3,847 | 2,3736 | 2,3699 |

En cuanto a las conclusiones generales y alineación con los objetivos, podemos decir que, a lo largo de las cuatro experimentaciones, comprobamos que un modelo base pre-entrenado y aplicar sucesivas rondas de fine-tuning permite explorar el equilibrio entre el recall y la precisión, o en otras palabras, hacer que el modelo tenga la capacidad de encontrar todas las instancias y evitar falsos positivos.

**Análisis de los resultados obtenidos. Reflexión sobre los retos encontrados y cómo proseguir:**

Con estas primeras experimentaciones vimos que el modelo sin congelamiento alcanzó el F1 más alto, pero luego cede terreno cuando se prioriza el recall, por lo que se introdujo metodologías en los siguientes experimentos que tuvieran tasas de aprendizaje diferenciados y un scheduler, por lo que se logró recuperar parte de la precisión y se mejoró la estabilidad de las métricas, aunque el F1 aun no llega al objetivo.

En todos los experimentos, el MAE y el RMSE se mantuvieron debajo de las referencias planteadas, confirmando que el modelo HerdNet, aun con fine-tuning, es un modelo altamente preciso para estimar conteos, superando la cota de error esperada.

En cuanto al F1 Score, se evidencia que ningún experimento alcanza aún el 83.5% de F1 deseado, pero la tendencia es creciente, por lo que se requiere seguir iterando y experimentando para poder llegar a la meta esperada, ya que el patrón creciente sugiere que la combinación de mayor variedad de datos, aumentos más avanzados y ajustes hiperparamétricos podría empujar el F1 sobre la barrera objetivo.

En mención del grado de alineación con los objetivos, en términos generales, podemos concluir por ahora que las metas de MAE y RMSE se superan holgadamente desde el experimento inicial. El F1 Score aún no alcanza el 83.5%, quedando a aproximadamente 3 puntos porcentuales según la estrategia. Por último, el comportamiento observado coincide con lo planteado en la propuesta de solución: usar HerdNet + fine-tuning es factible para mantener bajos errores de conteo y mejorar progresivamente el equilibrio entre el recall y la precisión.

Los retos iniciales que el grupo de trabajo presenció se dió principalmente en el procesamiento de las imágenes. Dado a su alta resolución y al peso de las mismas, no había sido posible procesarlas para ejecutar el pipeline de preprocesamiento, obligando al equipo a hacer metodologías de particionamiento. Esto no solo soluciona el problema de carga de data, sino que también se vio que las imágenes daban mejor detalle al modelo para facilitar su procesamiento y rendimiento, teniendo en cuenta que, ante el ojo humano, son imágenes donde no se logran ver o ubicar los animales de una manera fácil.

Otro reto significativo también consistió en el costo computacional para procesamiento del modelo en sus variaciones experimentales, en donde el grupo de trabajo ha tenido que hacer un gasto adicional para poder obtener acceso al consumo de unidades de cómputo de alto rendimiento para poder ejecutar estos modelos.

En conjunto, los cuatro experimentos iniciales muestran que la propuesta de solución cumple con los requisitos de error y está cerca de lograr el F1 esperado. Se aplicará un refinamiento exhaustivo en las prácticas de entrenamiento y la variedad de datos para alcanzar y superar el umbral esperado.

En relación con los retos identificados inicialmente para estas experimentaciones, el grupo de trabajo mantendrá metodologías de particionamiento de las imágenes para evitar problemas en el preprocesamiento de las mismas y así continuar con facilitar la provisión de imágenes al modelo que se entrene. Adicionalmente, se continuará con el gasto y consumo de las unidades de cómputo para continuar con el entrenamiento de los modelos en sus variedades experimentales, aunque en las siguientes experimentaciones, se generará con una muestra de imágenes (como se hizo en la experimentación 4) para tener un acercamiento del rendimiento de la experimentación y así ir entrenando el modelo de manera gradual en función de la cantidad de imágenes provisionadas.

**Próximos pasos:**

En cuanto a próximos pasos, se piensa hacer un preprocesamiento de data que incluya Data Augmentation como MixUp, CutMix, o transformaciones basadas en dominio para enriquecer el entrenamiento.

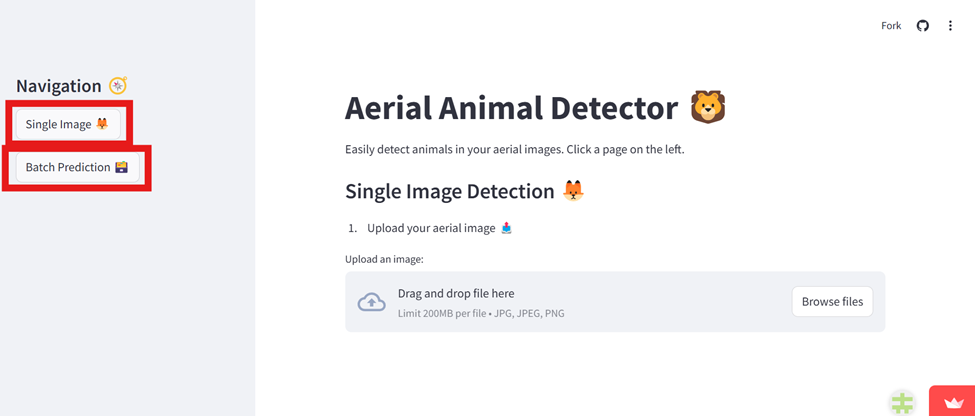
También se seguirá iterando en experimentar con más hiperparámetros, haciendo una búsqueda de hiperparámetros automatizada que nos ayude a guiarnos en la obtención de hiperparámetros que combinados mejoren las métricas obtenidas, usando Grid Search inicialmente en la tasa de aprendizaje, el weight\_decay y el tamaño del batch.

Adicionalmente, pero no como esencia, se añadirían módulos de atención o capas ligeras para reforzar la discriminación de las clases de animales con menor cantidad de data.

Por último, se piensa combinar instancias de HerdNet o modelos más simples para robustecer la precisión.

Es de anotar que, ante la gran cantidad de opciones para fine-tuning y variaciones de opciones en la edición de la arquitectura del modelo, los anteriores son pasos iniciales que se han pensado, pero es posible que no sean definitivos, dado que las variaciones en las decisiones pueden darse en el momento de generar las experimentaciones por seguir.

Por último, estamos trabajando en un front-end que facilite el uso de nuestro modelo, por medio de cargar imágenes en el front y que en el back se encuentre conectado nuestro modelo, de tal manera que una vez procesada la imágen, retorne los resultados obtenidos, a continuación incluimos una imágen del front-end:



El siguiente paso será hacer más pruebas para asegurar el funcionamiento correcto de nuestra aplicación.

**Bibliografía:**

* [Delplanque et al., 2022](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0045), [Eikelboom et al., 2019](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0050), [Kellenberger et al., 2017](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0095), [Kellenberger et al., 2018](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0100), [Kellenberger et al., 2019a](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0105), [Naudé and Joubert, 2019](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0185), [Peng et al., 2020](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0225), [Torney et al., 2019](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0275)
* [Gao et al., 2020](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0340)
* [Lempitsky and Zisserman (2010)](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0140)
* [Li et al., 2021](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0145), [Liu et al., 2018](https://www.sciencedirect.com/science/article/pii/S092427162300031X?via%3Dihub%22%20%5Cl%20%22b0175)
* <https://www.uniandes.edu.co/es/noticias/ingenieria/inteligencia-artificial-para-la-proteccion-del-amazonas>