

Monitoreo de vida salvaje mediante detección de objetos en diferentes tipos de imágenes

Antezana, Matias

*Ingeniería en Inteligencia Artificial
Universidad de San Andrés
Buenos Aires, Argentina
mantezana@udesa.edu.ar*

Giacometti, Mateo

*Ingeniería en Inteligencia Artificial
Universidad de San Andrés
Buenos Aires, Argentina
mgiacometti@udesa.edu.ar*

I. INTRODUCCIÓN

El monitoreo y conservación de la vida silvestre son actividades fundamentales para la preservación de los ecosistemas. En este contexto, la tecnología ha desempeñado un papel crucial al proveer herramientas avanzadas para la detección y seguimiento de animales en sus hábitats naturales. En particular, el uso de drones equipados con cámaras multiespectrales ha demostrado ser altamente efectivo, no solo por su capacidad de cubrir grandes áreas con costos reducidos, sino también por su potencial para integrar métodos automatizados basados en Inteligencia Artificial.

Este trabajo práctico a sido realizado con la finalidad de utilizar la conocida arquitectura de detección de objetos *YOLO* (You Only Look Once) para *detectar* y *clasificar* cada una de las *tres especies animales* (vacas, ciervos y caballos) presentes en el set de datos *Aerial Wildlife Image Repository* de la *Universidad de Mississippi*. Este mismo cuenta con una colección de *imágenes RGB*, *imágenes termales* y *coordenadas de los animales* en las imágenes etiquetadas por biólogos expertos.

Para cumplir con este objetivo, se entrenaron varios modelos basados en la arquitectura mencionada, utilizando los datos provistos por el conjunto de imágenes. Se exploraron distintas combinaciones de datos, implementando variaciones en la composición de los mismos y evaluando estrategias de fusión de resultados para mejorar la robustez y precisión del sistema de detección.

En las siguientes secciones, se describen en detalle los datos utilizados en las etapas de entrenamiento, validación y testeo, así como las diversas arquitecturas empleadas. Posteriormente, se presentan los resultados obtenidos junto con un análisis exhaustivo de los mismos. Finalmente, se establecen las conclusiones y se enumeran las referencias bibliográficas que sustentan y enriquecen el desarrollo de este trabajo.

II. CONJUNTO DE DATOS

II-A. Datos provistos

Como se mencionó anteriormente, los datos utilizados en esta investigación consisten en conjuntos de imágenes en formato *RGB* y su correspondiente *versión termal*, junto con

sus *labels* asociados. Estos labels incluyen la clase a la que pertenece cada animal en una imagen, las coordenadas del centro del *bounding box* (el rectángulo que delimita al animal dentro de la imagen) y las dimensiones del mismo. Tanto las coordenadas del centro como el ancho y alto del *bounding box* están expresados en términos relativos a las dimensiones de la imagen, utilizando valores normalizados entre 0 y 1. En total, el conjunto de datos está compuesto por 68 pares de imágenes *RGB* y termales correspondientes a vacas, 53 pares para ciervos y 43 pares para caballos.

II-B. Datos generados

A partir de los pares de imágenes *RGB* y termales, se generaron dos nuevos conjuntos de datos mediante la combinación y transformación de los canales de ambas imágenes (cada una compuesta por tres canales), con el objetivo de evaluar si la información proporcionada por ambos formatos podía complementarse para mejorar la precisión en la detección y clasificación.

■ **Imágenes HST:** La generación de imágenes *HST* comienza transformando las imágenes convencionales del espacio de color *RGB* al espacio *HSV* (Hue, Saturation, Value), que refleja de manera más fiel la percepción visual humana. Este espacio permite descomponer la información de la imagen en componentes clave: el *tono* (*H*), que indica el color dominante; la *saturación* (*S*), que representa la intensidad o pureza del color; y el *valor* (*V*), que define la luminosidad de la imagen. Para formar el espacio *HST*, se seleccionan las componentes de tono y saturación del espacio *HSV* y se combinan con el canal térmico (*T*), que proviene de las imágenes termales en escala de grises. Esta fusión genera un espacio de tres canales (*H*, *S* y *T*) que incorpora información sobre el color y la intensidad cromática de los objetos junto con sus características térmicas.

■ **Imágenes GST:** La representación *GST* emplea una estrategia diferente para integrar datos visuales y térmicos, poniendo énfasis en la luminancia y la intensidad del color, además de la información térmica. Este espacio se compone de tres canales: el canal *G*, que se obtiene al transformar la imagen *RGB* a escala de grises, conser-

vando la información de luminancia general; el canal *S*, que corresponde a la saturación extraída del espacio HSV, proporcionando detalles sobre la intensidad cromática; y el canal *T*, que incluye la información térmica en escala de grises, destacando las diferencias de temperatura en la escena.

A continuación, se presenta un ejemplo de cada uno de los tipos de imágenes utilizados en este trabajo:

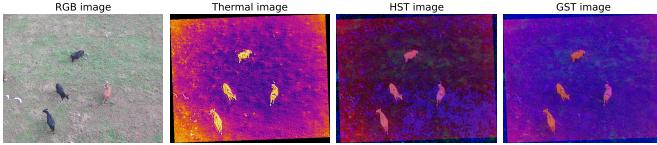


Figura 1. Tipos de imágenes.

III. METODOLOGÍA

III-A. Explicación del modelo YOLOv11

Para el entrenamiento de los modelos se utilizó la última versión de la serie **Ultralytics YOLOv11**. A continuación se muestra una representación de la arquitectura de YOLOv11:

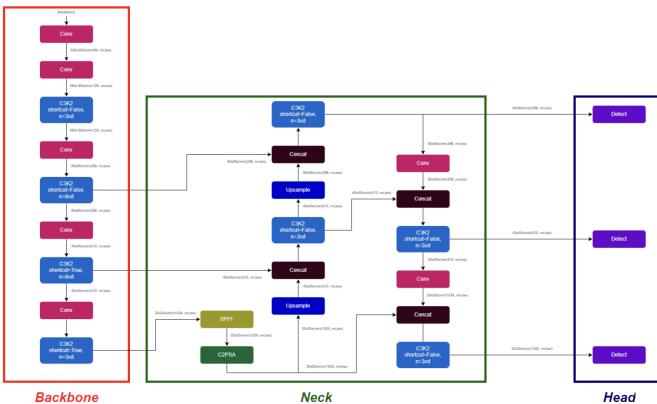


Figura 2. Arquitectura de YOLOv11.

Por un lado, la arquitectura cuenta con un *Backbone* la cual es responsable de extraer las características esenciales de las imágenes de entrada. Utiliza bloques convolucionales y procesa los *c* (canales), *h* (altura) y *w* (ancho) dados a pasando a través de una capa convolucional 2D seguida de una capa de normalización por Lotes 2D y finalmente con una función de activación *SiLU*. A su vez, el modelo posee una estructura conocida como *Neck* en el cual el modelo consolida y reina los mapas de características para las tareas posteriores. Por otro lado, al igual que en las versiones anteriores de YOLO, YOLOv11 utiliza una estructura *Head* de predicción multiescala para detectar objetos de distintos tamaños. Esta parte genera cuadros de detección para tres escalas diferentes (baja, media, alta) utilizando los mapas de características (siendo por lo general P3, P4 y P5) generados por las estructuras *Backbone* y *Neck*. De esta forma, se garantiza que los objetos pequeños se

detecten con mayor detalle (P3), mientras que los objetos más grandes se capturan mediante características de nivel superior (P5).

Con respecto a sus predecesores, este modelo ofrece nuevas mejoras:

- **Backbone basada en transformers:** Se utiliza un Backbone basado en transformers que captura dependencias de largo alcance y mejora la detección de objetos pequeños.
- **Una estructura del Head dinámico:** Logrando que se adapte a la complejidad de la imagen, optimizando la asignación de recursos para un procesamiento más rápido y eficiente.
- **Asignación de etiqueta dual:** Mejora la detección en objetos superpuestos y densamente empaquetados mediante un enfoque de asignación de etiquetas de uno a uno y de uno a muchos.
- **Convoluciones de kernels grandes:** Permiten una mejor extracción de características con menos recursos computacionales, lo que mejora el rendimiento general del modelo.
- **Autoatención parcial (PSA):** Aplica selectivamente mecanismos de atención a ciertas partes del mapa de características, mejorando el aprendizaje de la representación global sin aumentar los costos computacionales.
- **Menos cantidad de parámetros que sus predecesores:** Esto permite mejorar la velocidad y la precisión de los modelos generados.

Modelo	Velocidad (FPS)	Precisión (mAP)	Parámetros
YOLOv5	45FPS	56.8	44 millones
YOLOv10	50FPS	58.2	48 millones
YOLOv11	60FPS	61.5	40 millones

Tabla I
COMPARACIÓN EN VELOCIDAD, PRECISIÓN Y PARÁMETROS ENTRE YOLOV11 Y SUS PREDECESORES

III-B. División de los datos

Para la división de los datos, se asignó el 70 % de las imágenes al conjunto de entrenamiento, el 20 % al de validación y el 10 % al de prueba. Para garantizar una separación coherente, se desarrolló un *pipeline* que organiza los datos en estos tres conjuntos, asegurando que las versiones *RGB* y *termiales* de una misma imagen permanezcan siempre en el mismo conjunto y no se distribuyan entre varios. Además, se utilizó un valor de *seed* fijo (132) en la función de partición, lo que garantiza la reproducibilidad en la división de los datos para todos los modelos.

III-C. Búsqueda de hiperparámetros

Antes de realizar el entrenamiento de los modelos, se llevó a cabo una búsqueda de hiperparámetros con el propósito de encontrar aquellos que maximicen el desempeño de cada modelo. Para esto se utilizó el método **Tuner** de la librería de **Ultralytics** el cual emplea un *algoritmo genético de mutación* para optimizar los hiperparámetros, entre ellos se encuentra el de mutación el cual consiste en buscar localmente el espacio

de hiperparámetros aplicando pequeños cambios aleatorios a los ya existentes, produciendo nuevos candidatos para la evaluación.

A continuación se muestran los mejores parámetros encontrados para cada uno de los modelos:

Modelo	Epochs	Batch	Opt.	LR	Mom.	IoU
RGB	100	8	AdamW	0.01	0.937	0.7
THERM	100	8	AdamW	0.006	0.948	0.7
RGB-T	100	8	AdamW	0.005	0.910	0.7
HST	100	8	AdamW	0.01	0.937	0.7
GST	100	8	AdamW	0.01	0.940	0.7

Tabla II
PARÁMETROS DE CADA MODELO

III-D. Entrenamiento de YOLOv11

Para realizar el entrenamiento de los modelos y la búsqueda de hiperparámetros mencionada anteriormente, se utilizaron las **GPU T4 de NVIDIA**, proporcionadas por *Google Colab* y el espacio de almacenamiento de *Google Cloud*. Se partió de una versión preentrenada del conjunto de datos COCO, que cuenta con 330,000 imágenes y más de 80 categorías de objetos, como base para el modelo a entrenar.

En total, se desarrollaron seis modelos con diferentes enfoques de entrenamiento. Por un lado, se entrenaron modelos utilizando únicamente las imágenes RGB como entrada y, por otro, modelos que emplearon únicamente su versión termales. Además, se implementó un modelo que utiliza ambos tipos de imágenes (RGB y termales) como entrada conjunta. Adicionalmente, se desarrollaron dos modelos que emplean las imágenes HST y GST, mencionadas en el apartado de *Datos generados*. Finalmente, se aplicó la metodología de Late Fusion, cuya explicación se presenta en la siguiente sección.

III-E. Combinación de resultados de modelos - Late Fusión Approach

La **Late Fusión** es un enfoque de fusión de datos en el que se procesan de manera independiente los datos de distintas fuentes, y la fusión ocurre en una etapa posterior, generalmente en el nivel de las salidas de los modelos. A diferencia de otros métodos de fusión temprana, donde los datos se combinan antes del procesamiento, en Late Fusion, cada fuente de información se procesa de forma separada y luego se integran los resultados obtenidos de cada fuente. En este caso, se propone la combinación de las predicciones de dos modelos entrenados con diferentes tipos de imágenes: RGB y termales.

Los resultados de detección obtenidos de ambas redes son procesados por un componente propuesto, no entrenable, denominado **Sistema de Emparejamiento de Predicciones (PMS)**, cuya función es generar una única salida consolidada a partir de los resultados de ambas ramas de procesamiento. Para lograrlo, el PMS considera como entradas el valor del parámetro de *intersección sobre unión (IoU)* entre las *bounding boxes* propuestas por ambos modelos, así como las

probabilidades asignadas por cada modelo a la pertenencia del objeto detectado a una clase específica. Una detección se considera válida únicamente si la clase detectada por ambos modelos coincide, las probabilidades de pertenencia a esa clase superan el umbral de 0.6 en ambos casos, y el valor de IoU entre las *bounding boxes* es superior a 0.45. De esta manera, el PMS fusiona las predicciones de manera precisa y confiable.

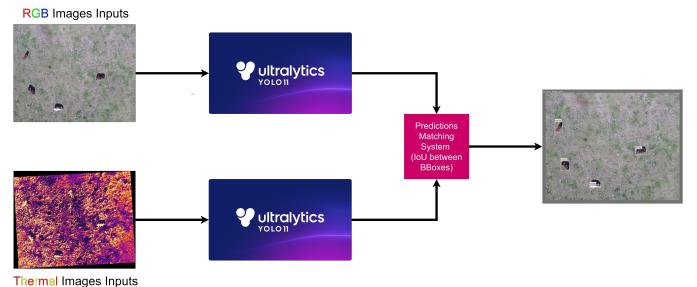


Figura 3. Late Fusión aplicado a YOLOv11. Se combinan las predicciones de un modelos entrenado con imágenes RGB y otro entrenado con imágenes termales.

La principal ventaja de la Late Fusión es que permite que los modelos trabajen con los datos de forma más especializada antes de combinar los resultados, lo que puede mejorar la precisión y robustez de las predicciones. Este enfoque es útil en escenarios donde diferentes tipos de datos aportan información complementaria que es mejor procesada por redes distintas antes de ser fusionada para una decisión final.

IV. RESULTADOS

A continuación, se presentan los resultados obtenidos mediante la utilización de los diferentes modelos propuestos. Cabe aclarar que todas la métricas obtenidas obtuvieron utilizando el *set de testeo* de cada modelo en particular.

IV-A. Evaluación de métricas de rendimiento en clasificación y detección

IV-A1. Ejemplos de detección

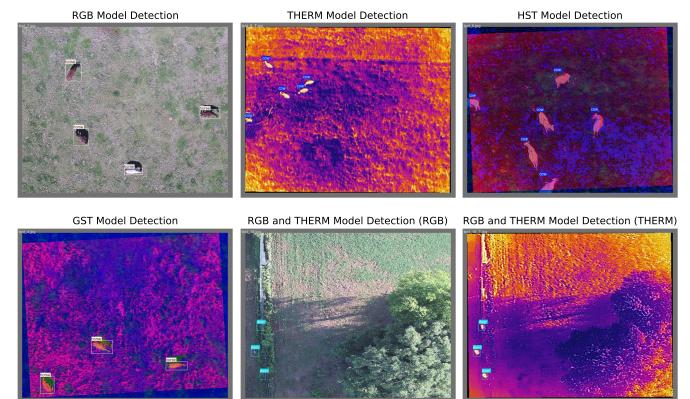


Figura 4. Ejemplos de detección de cada tipo de modelo realizado en el trabajo.

Como se aprecia en estos ejemplos, cada modelo logra identificar correctamente la clase a la que pertenece cada animal presente en las imágenes, al tiempo que define con precisión las dimensiones del bounding box, delimitando claramente a cada animal.

IV-A2. Matrices de confusión

Una matriz de confusión es una herramienta clave para evaluar el desempeño de un modelo, ya que permite comparar las predicciones realizadas con las etiquetas reales. Cada celda de la matriz representa el número de instancias clasificadas en una categoría específica en relación con su clase verdadera, proporcionando una visión detallada de los aciertos y errores del modelo. A continuación, se presentan las matrices de confusión correspondientes a cada modelo, obtenidas a partir de su desempeño sobre el set de test:



Figura 5. Matriz de confusión del modelo entrenado con imágenes RGB.

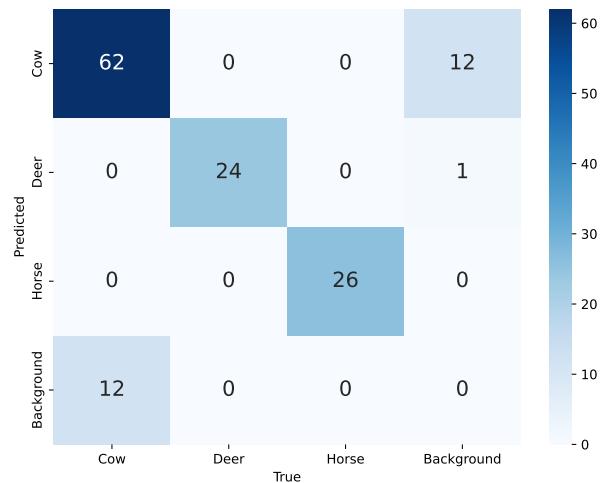


Figura 7. Matriz de confusión del modelo entrenado con imágenes RGB y termales.

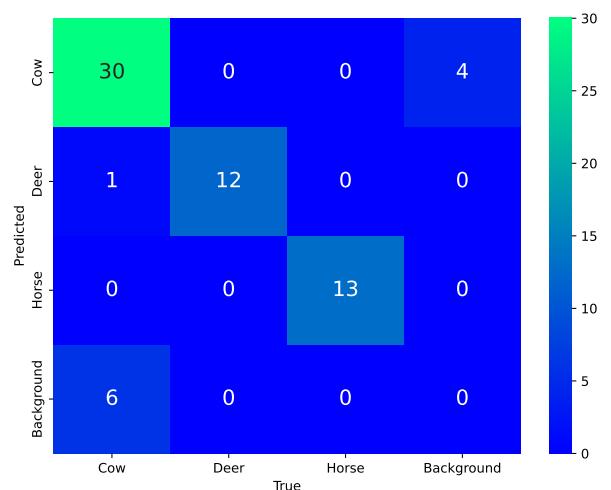


Figura 8. Matriz de confusión del modelo entrenado con imágenes HST.

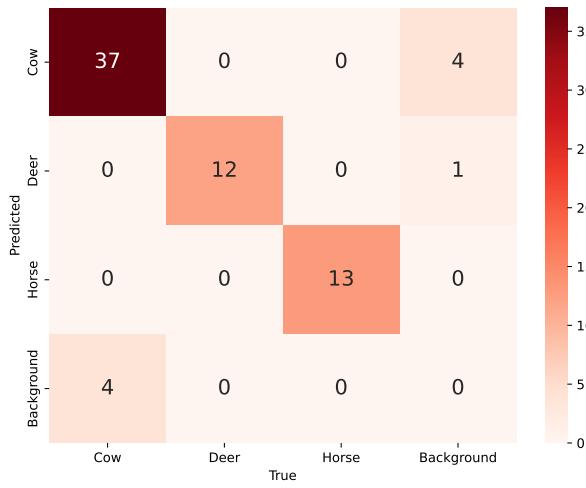


Figura 6. Matriz de confusión del modelo entrenado con imágenes termales.

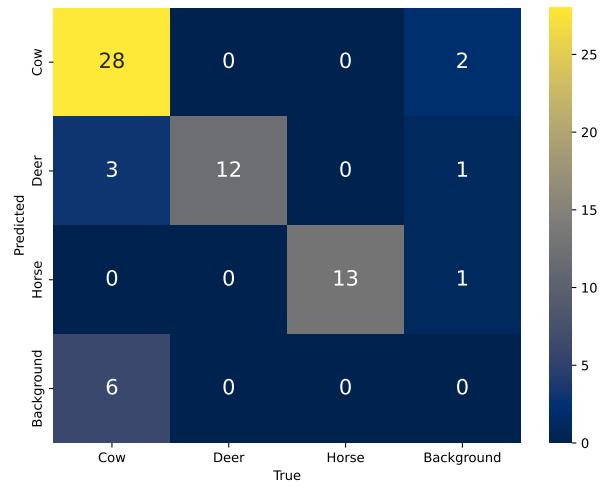


Figura 9. Matriz de confusión del modelo entrenado con imágenes GST.

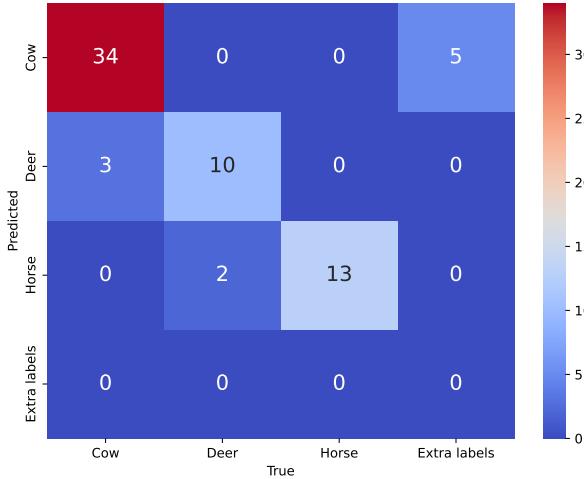


Figura 10. Matriz de confusión del modelo Late Fusion.

Como se puede apreciar, los modelos de detección mostraron un rendimiento variable según la clase analizada. La clase *cow* presentó un desempeño superior, logrando una identificación más precisa de los objetos, aunque se observaron algunas confusiones al clasificar regiones del fondo como pertenecientes a esta categoría. Por otro lado, las clases *deer* y *horse* mostraron un rendimiento moderado, con una menor precisión y ciertas confusiones, particularmente con la clase *background*. Esta última resultó ser la más problemática, ya que los modelos tuvieron dificultades para diferenciar entre objetos y regiones sin interés, lo que generó errores recurrentes en las predicciones.

Estos resultados sugieren la necesidad de optimizar la capacidad de discriminación entre objetos y fondo. Para ello, se recomienda mejorar el balance de los datos de entrenamiento, dado que, como se observó al inicio del informe, existe un mayor volumen de imágenes de vacas en comparación con las de caballos y ciervos. Además, sería beneficioso aumentar la cantidad total de datos disponibles para garantizar una representación más equitativa y robusta de todas las clases.

IV-A3. Métricas de Rendimiento

En el siguiente apartado, se presentan una serie de métricas promedio obtenidas de la evaluación de los modelos sobre sus respectivos conjuntos de testeo. Estas métricas incluyen: *Precision* (proporción de predicciones correctas), *Recall* (capacidad del modelo para detectar correctamente los objetos reales), *F1-Score* (promedio armónico entre precisión y recall), *mAP₅₀* (precisión promedio con un umbral de IoU de 0.5) y *mAP₅₀₋₉₅* (precisión promedio calculada en un rango de IoU de 0.5 a 0.95).

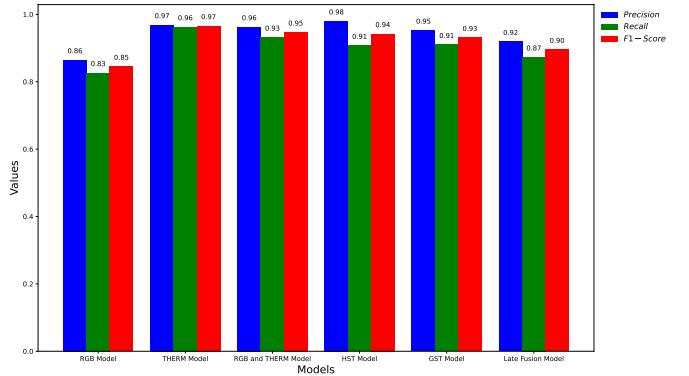


Figura 11. Métricas de Precision, Recall y F1-Score de cada modelo realizado.

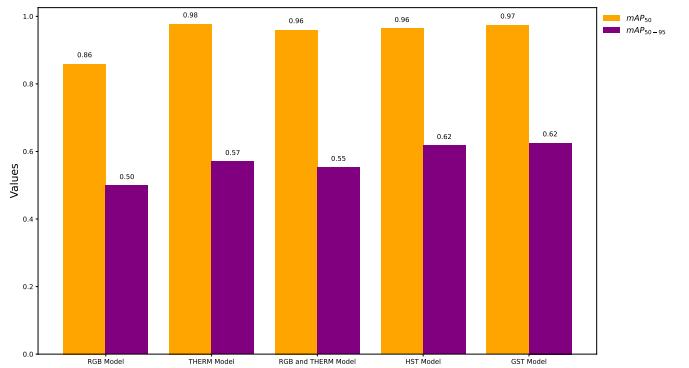


Figura 12. Mean Average Precision al 50 % y desde 50 % a 95 % de cada modelo realizado.

IV-B. Análisis de rendimiento de cada modelo

En base a la información presentada en la sección anterior, se procede a analizar el rendimiento individual de cada modelo:

- **RGB Model:** Es el modelo que presenta las métricas mas bajas, con una *precision* notable de 0.86 y un *recall* de 0.83, lo que significa que es bastante eficiente para identificar correctamente las especies en la mayoría de los casos, aunque no detecta todas las especies reales (un *recall* ligeramente menor). El *F1-Score* de 0.85 es equilibrado, lo que indica un rendimiento razonable en términos de la media entre precisión y recall. En términos de *mAP₅₀*, el modelo logra un 0.86, lo que sugiere que es eficiente en la detección de objetos a umbrales más bajos de IoU. Sin embargo, su rendimiento cae considerablemente en *mAP₅₀₋₉₅*, con un valor de 0.50, indicando que no mantiene un buen desempeño cuando se requiere una superposición más estricta de las bounding boxes. Esto sugiere que el modelo tiene dificultades con la localización precisa de los objetos, lo cual podría ser una limitación inherente a las imágenes RGB en comparación con otros formatos (como las termales).

- **THERM Model:** Destaca como el mejor en cuanto a precision y recall. La *precision* de 0.97 indica que casi todas las predicciones hechas por el modelo son correctas,

y el *recall* de 0.96 muestra que el modelo identifica correctamente la mayoría de los animales. El *F1-Score* de 0.97 refleja el equilibrio perfecto entre precisión y recall, lo que lo convierte en un modelo de alto rendimiento para esta tarea. En términos de mAP_{50} , el modelo también alcanza un valor excepcional de 0.98, lo que confirma que el modelo térmico tiene un rendimiento sobresaliente incluso con umbrales bajos de IoU. Sin embargo, el mAP_{50-95} es más bajo, con un valor de 0.57, lo que sugiere que, aunque el modelo realiza muy bien en detección a IoU más bajos, tiene dificultades para mantener este rendimiento cuando se exige una localización más precisa.

- **RGB-THERM Model:** El modelo entrenado con imágenes RGB y termales tiene un rendimiento muy similar al *THERM Model*, con *precision* y *recall* cercanos a 0.96 y 0.93, respectivamente, lo que indica que no se pierde mucha información al combinar ambas modalidades. Sin embargo, el *F1-Score* es ligeramente más bajo (0.95) en comparación con el modelo térmico solo. En cuanto a mAP_{50} , el modelo combinado alcanza 0.96, lo que sigue siendo alto, pero no mejora significativamente sobre el modelo térmico solo (que tiene 0.98). En términos de mAP_{50-95} , este modelo tiene un valor de 0.55, lo que está en línea con el modelo termal, indicando que la combinación de modalidades no mejora significativamente la capacidad de localización precisa de los objetos.

- **HST Model:** El modelo HST tiene una *precision* muy alta (0.98), lo que significa que el modelo hace pocas predicciones incorrectas. Sin embargo, el *recall* (0.91) es un poco más bajo en comparación con otros modelos, lo que sugiere que puede estar perdiendo algunas detecciones. A pesar de esto, el *F1-Score* sigue siendo bastante alto (0.93), indicando un buen balance entre *precision* y *recall*. En términos de mAP_{50} , el modelo tiene un valor de 0.96, lo que indica que es eficiente para detección a umbrales bajos de IoU. En mAP_{50-95} , el modelo mejora con respecto a los modelos *RGB* y *RGB+Termal*, alcanzando 0.62, lo que indica que tiene un mejor rendimiento en localización precisa de los objetos en comparación con los modelos anteriores.

- **GST Model:** El modelo GST muestra un rendimiento bastante similar al modelo HST, con *precision* y *recall* de 0.95 y 0.91, respectivamente. Al igual que el modelo HST, el *F1-Score* de 0.93 indica un buen balance. En cuanto a mAP_{50} , el modelo muestra un excelente resultado de 0.97, similar al modelo termal. Sin embargo, su mAP_{50-95} es de 0.62, que es igual al modelo HST, lo que sugiere que el GST tiene una capacidad similar para detectar y localizar objetos de forma precisa.
- **Late Fusion Model:** El modelo de Late Fusion tiene un buen rendimiento en términos de *precision* (0.93) y *recall* (0.92), aunque su *F1-Score* de 0.90 es más bajo que el de los modelos termales, lo que indica que no ha sido tan eficaz en equilibrar la *precision* y el *recall*.

V. CONCLUSIONES

En base a los resultados obtenidos en la sección anterior, se pueden llegar a las siguientes conclusiones:

- El modelo basado exclusivamente en **imágenes termales** se destaca como el más eficaz, logrando la mayor precisión y recall. Esto sugiere que las diferencias de temperatura entre especies son características clave para la clasificación en este dataset.
- Las **imágenes RGB** parecen ofrecer información complementaria, pero no esencial, dado que su **combinación** con las **imágenes termales no mejora significativamente los resultados**. De hecho, en algunos casos, incluso podría introducir ruido adicional.
- Los modelos basados en transformaciones (**HST** y **GST**) también logran **buenos resultados**, pero no superan el rendimiento del modelo termal puro. Esto indica que, aunque útiles, estas transformaciones no capturan información crucial para este problema en comparación con la modalidad termal.
- La técnica de **Late Fusion** no ofrece **beneficios significativos**, lo cual podría estar relacionado con una subóptima integración de los modelos o con redundancia en la información.

REFERENCIAS

- [1] K. Roszyk, M. R. Nowicki, and P. Skrzypczyński. *Adopting the YOLOv4 Architecture for Low-Latency Multispectral Pedestrian Detection in Autonomous Driving*. Sensors, vol. 22, no. 3, pp. 1–19, 2022. <https://www.mdpi.com/1424-8220/22/3/1082>
- [2] S. Liang, H. Wu, L. Zhen, Q. Hua, S. Garg, G. Kaddoum, M. M. Hassan and K. Yu. *YOLO: Real-Time Intelligent Object Detection System Based on Edge-Cloud Cooperation in Autonomous Vehicles*. Journal of Advanced Transportation, vol. 2022, pp. 1–15, 2022. <https://arxiv.org/abs/2205.14942>
- [3] Roboflow Blog. *YOLOv11: How to Train Custom Data*. <https://blog.roboflow.com/yolov11-how-to-train-custom-data/>
- [4] Ultralytics. *Ultralytics YOLO: Hyperparameter Tuning*. <https://docs.ultralytics.com/guides/hyperparameter-tuning/#introduction>
- [5] N. Rao. *YOLOv11 Explained: Next-Level Object Detection with Enhanced Speed and Accuracy*. Medium, 2024. <https://medium.com/@nikhil-rao-20/yolov11-explained-next-level-object-detection-with-enhanced-speed-and-accuracy-2dbe2d376f71>
- [6] Analytics Vidhya. *YOLOv11: El siguiente paso en la detección de objetos en tiempo real*. <https://www.analyticsvidhya.com/blog/2024/10/yolov11/>