E3.1. Algoritmos inteligentes Explicables para la identificación y predicción de aves

GOBIERNO DE ESPAÑA
VICEPRESIDENCIA TERCERA DEL GOBIERNO
MINISTERIO PARA LA TRANSICIÓN ECOLÓGICA Y EL RETO DEMOGRÁFICO
Fundación Biodiversidad
Plan de Recuperación, Transformación y Resiliencia
Financiado por la Unión Europea
NextGenerationEU

1. Introducción

1.1 Objetivo

Este documento corresponde al entregable: **E3.1 - Algoritmos inteligentes Explicables para la identificación y predicción de aves**

Refleja los trabajos realizados y los resultados alcanzados durante la ejecución de la actividad:

A3.1 Redes Neuronales Recurrentes Convolucionales para la detección acústica de aves, A3.2 Redes Convolucionales para el procesamiento de mapas de sensibilidad de vida silvestre y A3.3 Redes Neuronales bayesianas Explicables para la predicción de los efectos acumulativos en las aves y sus hábitats.

Esta tarea se encuadra dentro de LA ACTIVIDAD A3 y la línea de investigación:

A3. Machine Learning para la detección y monitorización de las aves cuyo objetivo se centra en investigar en algoritmos híbridos de Inteligencia Artificial Explicable para la monitorización y detección de aves y el procesamiento de mapas de sensibilidad, así como modelos predictivos que permitan identificar bandadas y estimar efectos acumulativos.

1.2 Alcance

Este documento se encuentra en la versión 1.0, y es la última revisión de los trabajos realizados en la A3., la investigación aquí plasmada trata de presentar el trabajo realizado para la consecución de los objetivos de la tarea de investigar en algoritmos híbridos de Inteligencia Artificial Explicable para la monitorización y detección de aves y el procesamiento de mapas de sensibilidad.

2 Introducción

El contenido de este entregable incluye todos los estudios, entrenamientos y pruebas realizadas para implementar un sistema de IA automático para la detección de aves y poder clasificar especies, y también como se han realizado mapeados del cielo y el tratamiento de esa información captada para diseñar visualizaciones.

Esta primera fase se divide en 2 grandes apartados. El primero de ellos es la monitorización visual de aves, y el segundo es la monitorización acústica de las mismas. En ambos se incluyen explicaciones sobre el estado del arte, la arquitectura implementada y los datos utilizados para entrenar los distintos modelos. Así mismo, en el apartado de monitorización visual se incluye un subapartado donde se trata el tema de la cámara empleada y la calibración de esta.

La energía eólica es una de las fuentes de energía renovables y limpias que más se están usando en la actualidad para luchar contra el cambio climático, la reducción de gases de efecto invernadero y conseguir la autonomía energética de los distintos países. No obstante, los parques eólicos constituyen un gran peligro para las aves autóctonas, pues corren el riesgo de chocar con los aerogeneradores, así como la pérdida de su hábitat y el desplazamiento de este.

Es por ello por lo que es necesario poder identificar y clasificar las aves existentes en un espacio geográfico para cuantificar el riesgo que corren en caso de crear un parque eólico en esa localización. Para ello, se han creado una serie de modelos de inteligencia artificial para poder identificar y clasificar las aves mediante sus imágenes y los sonidos que emiten.

Para la satisfactoria realización del proyecto, se han desarrollado 3 algoritmos: uno mediante visión artificial con un modelo de aprendizaje automático supervisado para poder segmentar las aves en una imagen. El segundo algoritmo usa modelos (supervisados) para clasificar las imágenes de las aves detectadas, y el último, también supervisado, se encarga de clasificar las aves según el sonido que emiten. Con el uso conjunto de algoritmos de segmentación y clasificación, se pretende crear mapas de calor en función de las aves detectadas. Además se pretende integrar esta información con información proveniente de otras fuentes de datos para construir mapas más completos y certeros.

3 Redes Neuronales Recurrentes Convolucionales para la detección acústica de aves

En esta sección se documenta toda la información previa al desarrollo de los distintos modelos relacionados con el sonido de las aves. Por otra parte, se define la arquitectura y funcionamiento del módulo encargado de la monitorización acústica de aves, así como el proceso de desarrollo seguido.

3.1 Estado del arte

En esta sección se procede a hacer un repaso sobre el estado del arte relacionado con la monitorización acústica de las aves. Cabe destacar que la segmentación de sonidos, a diferencia de la segmentación de imágenes, está muy poco desarrollada, y por eso se hará más énfasis en la clasificación de sonidos, y no tanto en su segmentación.

3.1.1 Clasificación de sonidos de aves

El sonido que las aves emiten, junto con su forma, tamaño y colores, es una de las características más distintivas de estos seres vivos. No obstante, primero es necesario hacer un preprocesamiento muy exhaustivo y específico de dichos sonidos para obtener características que puedan servir para entrenar un modelo de aprendizaje profundo que pueda clasificar los sonidos.

El sonido es una onda física producida por las diferencias de presión en el aire. Hay muchas maneras distintas de procesar los sonidos para entrenar modelos de aprendizaje profundo. No obstante, dependiendo del dominio del problema, unas características aportarán más información que otras, mejorando el aprendizaje del modelo. De nuevo, este proceso de extracción de características de sonidos es un proceso de prueba y error.

Las características que más comúnmente se utilizan son los espectrogramas y los cepstrums, como por ejemplo los Coeficientes Cepstrales de la frecuencia de Mel (MFCC), Coeficientes Cepstrales de la Frecuencia Gammatone (GTCC) o los Coeficientes Cepstrales de Predicción Lineal (LPCC). En los artículos [1], [2] se hace un resumen de los cepstrums más usados hasta la fecha para la clasificación de aves. Dependiendo de las características que se extraigan, así como del procesamiento posterior que se les realicen, se entrenan unos modelos u otros. Las características extraídas son matrices de 2 dimensiones para los espectrogramas, es decir, que

se pueden interpretar como imágenes con un solo canal de color. En el caso de MFCC, GTCC y LPCC son vectores de coeficientes.

En el caso de que tras obtener las características, se haga un aplanado de estas (se transforma la matriz de 2 dimensiones en un vector de 1 dimensión apilando horizontalmente las filas de la matriz), se pueden entrenar modelos como máquinas de vector soporte, árboles de decisión como random forest o redes fully connected para generalizar el concepto. En el artículo [20] se hace uso de este método, usando una máquina de vector soporte para hacer la clasificación. Si por el contrario no se hace el aplanado, se utilizan técnicas de aprendizaje profundo, normalmente basadas en CNN como las EfficientNet o basadas en transformers como los Visual Transformers.

Para seleccionar un método u otro, es crucial tener en cuenta primero el número de características extraídas de los sonidos, y segundo el número de datos disponibles para entrenar los algoritmos. En caso de tener pocos datos con pocas características se recomienda el uso del primer método, ya que se usan modelos más simples con menos parámetros a aprender, mientras que, si se disponen de muchos datos con muchas características, los modelos de visión artificial suelen presentar mejores resultados.

3.1.2 Segmentación de sonidos de aves

Las primeras aproximaciones para segmentar sonidos (no solo de aves) consiste en detectar aquellas porciones del audio con un nivel de intensidad (decibelios) inferior a la media de todo el audio. De esta forma se detectan los silencios relativos del audio, y se segmenta en base a esas zonas. No obstante, aunque puede presentar buenos resultados en distintos dominios de aplicación, no es una técnica que generaliza bien, pues solo funciona cuando hay porciones de audio muy diferentes por momentos de silencio. En caso de que existan clases que se solapan (ocurren al mismo tiempo, o no existe un silencio entre ellas) este método no obtiene buenas métricas.

El hecho anterior, sumado a las nuevas técnicas de aprendizaje profundo, han llevado a la creación de un modelo de segmentación de audio con deep learning, llamado YOHO, (‘You Only Hear Once’ en inglés) inspirado en el modelo YOLO usado en la segmentación de imágenes, tal y como se ha explicado en el apartado de estado del arte de monitorización visual. En el artículo [22] se define dicho modelo, así como su funcionamiento.

3.2 Arquitectura

La innovación principal en cuanto al procesamiento de audio ha tenido lugar gracias al diseño, implementación y refinamiento de la arquitectura propuesta. Además, su efectividad ha sido probada gracias a una evaluación y comparación rigurosa de métricas frente al estado del arte. A nivel conceptual se propone un cambio de paradigma de cara a clasificar sonidos (especies de aves según su canto). El nuevo paradigma propuesto altera el flujo clásico que se seguía en este tipo de tareas; es decir, la extracción de características ya no es un módulo aislado, sino que ahora se encuentra integrada junto al modelo. De esta manera, se ataca el principal cuello de botella que limitaba las capacidades de cómputo y rendimiento (el uso de espectrogramas). El método tradicional presenta tres fases muy bien diferenciadas: Ingesta del audio, extracción de características (MFCC, espectrograma, etc) y clasificación; como contrapartida, el enfoque propuesto utiliza una CNN unidimensional junto a transformadores para substituir a la extracción de características clásica.

Mediante este nuevo enfoque se evita el cálculo previo del espectrograma, lo que trae como consecuencia una reducción del tiempo de inferencia en un 3.5x y de las operaciones lógicas en un 25%. Esto no solo ha permitido mantener un rendimiento competitivo con datasets de altas prestaciones, sino que además ha permitido superar a los métodos tradicionales donde las clases minoritarias abundan. En definitiva, se ha descubierto una nueva arquitectura ligera optimizada para dispositivos edge —de bajo consumo energético y sin GPU —.

La arquitectura en cuestión a nivel conceptual se puede observar en el siguiente diagrama:

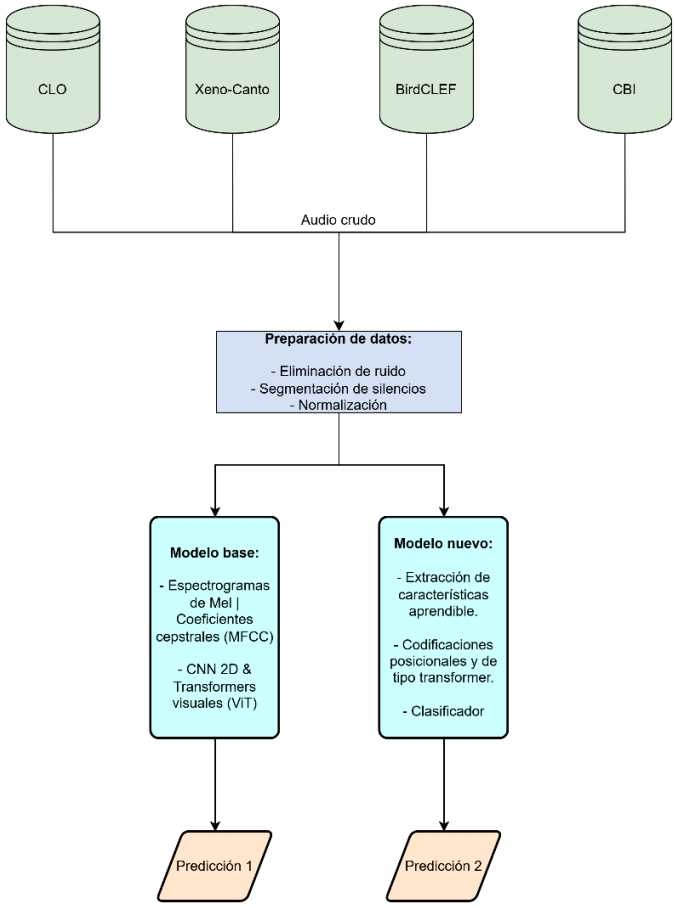


Fig.1. Arquitectura inovadora conceptual

Para poder justificar la mejora que trae consigo el modelo propuesto, ha sido sometido a rigurosas pruebas de evaluación. Algunas de las métricas utilizadas durante su evaluación son:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

$$Top5Accuracy = \left(\frac{1}{N}\right) * \sum_{1}^{N} \mathbb{1}(y_i \in Top5(\hat{y}_i))$$

La métrica Top-5 Accuracy evalúa la capacidad del modelo para incluir la clase correcta entre las cinco predicciones con mayor probabilidad. En esta expresión, Nrepresenta el número total de muestras, $y_i$la etiqueta real de la muestra iy Top5$(\hat{y}_i)$el conjunto de las cinco clases más

probables estimadas por el modelo. Se considera una predicción correcta si la clase real se encuentra dentro de ese conjunto. Esta métrica resulta especialmente útil en problemas de clasificación multiclase con gran número de categorías, ya que proporciona una visión más flexible del rendimiento global del modelo y complementa la medida tradicional de Top-1 Accuracy.

$$F1 = 2 * \frac{(Precision * Recall)}{(Precision + Recall)}$$

$$MACs = \sum_{1}^{L} (k_l * s_l * c_l)$$

El número de operaciones Multiply–Accumulate (MACs) representa una estimación del coste computacional total del modelo durante la inferencia. Cada operación MAC equivale a una multiplicación seguida de una suma, y constituye la unidad básica de cálculo en redes neuronales. En la expresión anterior, Ldenota el número total de capas, $k_l$el número de pesos o parámetros del filtro en la capa l, $s_l$el número de salidas generadas y $c_l$el número de canales de entrada. Un menor número de MACs implica un modelo más eficiente en términos de operaciones aritméticas y, por tanto, más adecuado para despliegue en dispositivos con recursos limitados (edge computing).

$$InferenceTime = \frac{T_{Total}}{N_{Samples}}$$

El Inference Time o tiempo medio de inferencia indica la duración promedio necesaria para clasificar una única muestra. Se obtiene dividiendo el tiempo total de procesamiento T_{total}entre el número total de muestras inferidas $N_{samples}$. Esta métrica evalúa la eficiencia temporal del modelo, considerando tanto el cálculo de características (como el espectrograma en arquitecturas tradicionales) como la inferencia interna en las capas del modelo. Valores bajos de Inference Time reflejan una mayor viabilidad para la ejecución en tiempo real y en plataformas embebidas de baja potencia.

3.2.1 Datos

Para entrenar el modelo de clasificación de audio, se hará uso de los datos públicos del dataset 'Xeno-canto' [3], CLO-43SD-AUDIO (CLO), BirdCLEF 2024 (BC24) y Cornell Birdcall Identification (CBI). Dichos datasets cuentan con una gran cantidad de grabaciones acústicas de sonidos de aves de todo el mundo. Para obtenerlos, se pueden descargar directamente por la página web [4] o mediante su API [5]. En la propia página web se hace una explicación exhaustiva del proceso de descarga de dichos audios.

Por cada uno de los audios, se ofrece una gran cantidad de información (coordenadas de la grabación, fecha, autor etc) pero solo se escogieron los audios en crudo (formato .wav) para entrenar el modelo de clasificación de sonidos.

Entre las 5 fuentes de datos, se obtuvieron un total de 62.152 audios en crudo y 151.141 auidos segmentados. No obstante, en cada uno de esos audios hay una gran cantidad de cantos del ave. Es por ello, que en el preprocesamiento de dichos audios, se segmentan estos audios para obtener cada uno de esos cantos individualmente. Tras realizar dicha segmentación, se obtienen un total de 18,802 audios, pertenecientes a 135 especies diferentes de aves.

3.3 Diseño y entrenamiento de los algoritmos para la monitorización acústica de aves.

El objetivo principal del módulo de sonido es la clasificación de diferentes especies de aves. Para ello se utilizarán los dataset de las fuentes mencionadas anteriormente, pero el estudio se llevará a cabo principalmente a través de los datos que ofrece XenoCanto desde su API. Los datos en bruto primeramente son extraídos directamente desde XenoCanto, después se aplica un pipeline de preprocesamiento donde se realiza una segmentación basada en silencios para conseguir la uniformidad de la longitud de los audios. Dentro del objetivo principal podemos englobar una serie de subobjetivos donde se destaca la innovación y la revisión sistemática del estado del arte actual del procesamiento de audio aplicado a la clasificación de cantos de aves. Los métodos utilizados en el estado del arte actual incluyen coeficientes cepstrales de mel, espectrogramas de mel, espectrogramas simples o múltiples características extraídas a partir de los espectrogramas de mel. En un ámbito tan asentado como el procesamiento del audio, innovar no es tarea sencilla, es por ello por lo que se ha tenido que acudir a términos como la eficiencia para tratar de aportar valor en este ámbito. Las aplicaciones más comunes del procesamiento de audio incluyen el uso de espectrogramas de mel junto con potentes modelos de aprendizaje transferido como EfficientnetB0. Sin embargo, la complejidad temporal (tiempo de entrenamiento) y la complejidad computacional (FLOPS y MAC) de este tipo de modelos es elevada, aunque sus buenas métricas lo avalen. Se pretende innovar en el procesado de audio a través de la construcción y el entrenamiento de un modelo mucho más liviano en términos de complejidad temporal y computacional que mantenga en la medida de lo posible las métricas que presentan los modelos de aprendizaje por transferencia. A continuación, se expondrán todas las fases por las que ha pasado este proceso de innovación hasta delimitar el alcance de la innovación y por último lograr su implementación y desarrollo

3.3.1 Métricas

En primer lugar, se encuentra la fase de tratar de obtener buenas métricas sin realizar un análisis exhaustivo del estado del arte. En esta primera fase se probó con diferentes modelos hasta dar con una solución efectiva. Se comenzó realizando una clasificación del tipo binaria entre dos especies por medio de una red neuronal convolucional montada de dos dimensiones (CNN 2D). Los resultados obtenidos fueron satisfactorios con un 97% de accuracy como se puede observar en la figura 1:

Reporte de Clasificación:

	precision	recall	f1-score	support
major	0.99	0.96	0.97	240
naturalis	0.96	0.99	0.98	240
accuracy			0.97	480
macro avg	0.98	0.98	0.97	480
weighted avg	0.98	0.97	0.97	480

Fig.2. Métricas 1

Siguiendo con el mismo objetivo en mente se trató de generalizar el buen funcionamiento del modelo a más clases, incorporando un total de 20 clases a clasificar. De nuevo se utilizó una red neuronal convolucional montada de dos dimensiones (CNN 2D) pero esta vez se alimentó con

muestras de hasta 20 clases diferentes. Los resultados no fueron tan brillantes como el anterior diseño, pero se obtuvo un 89% de accuracy como se puede observar en la figura 2:

Reporte de Clasificación:

	precision	recall	f1-score	support
Burhinus oedicnemus	0.94	0.84	0.88	55
Cettia cetti	0.86	0.80	0.83	55
Chloris chloris	1.00	0.85	0.92	55
Curruca melanocephala	0.98	0.87	0.92	55
Emberiza calandra	1.00	0.95	0.97	55
Emberiza cirlus	0.84	0.98	0.91	55
Erithacus rubecula	0.80	0.96	0.88	55
Fringilla coelebs	0.96	0.82	0.88	55
Gallinula chloropus	0.83	0.80	0.81	55
Himantopus himantopus	0.80	0.85	0.82	55
Larus michahellis	0.96	0.93	0.94	55
Luscinia megarhynchos	0.81	0.78	0.80	55
Parus major	0.76	0.80	0.78	55
Phylloscopus bonelli	0.96	0.91	0.93	55
Phylloscopus ibericus	0.87	0.95	0.90	55
Serinus serinus	0.95	0.98	0.96	55
Sonus naturalis	0.92	1.00	0.96	55
Sylvia atricapilla	0.88	0.95	0.91	55
Troglodytes troglodytes	0.96	0.95	0.95	55
Turdus merula	0.86	0.89	0.88	55
accuracy			0.89	1100
macro avg	0.90	0.89	0.89	1100
weighted avg	0.90	0.89	0.89	1100

Fig.3. Métricas 2

Cabe destacar que tanto para la clasificación binaria como para la clasificación multiclase se utilizaron espectrogramas de mel.

3.3.2 Algoritmos utilizados

Debido a las necesidades específicas del proyecto IA4BIRDS resultó más conveniente una adaptación de un modelo existente que la construcción del modelo desde cero como se venía haciendo en apartados anteriores. El módulo al que haremos referencia para basarnos en la adaptación será un módulo de segmentación de vídeos y extracción de audios.

Se modificaron los archivos audio_augmentator.py, clasification_module.py, configuration_py y utils.py para garantizar el cumpliendo con la estructura del proyecto mientras que se seguían los principios de diseño modular. Las principales diferencias son las siguientes: Desde clasification_module.py se modificaron parámetros específicos tales como batch_size, replicas o epochs ya que previamente venían siendo modificados en el archivo configuration.py. Otras configuraciones que son realizadas son la adaptación de debug, augment, training_plot, wandb, steps_per_epoch y validation_steps.

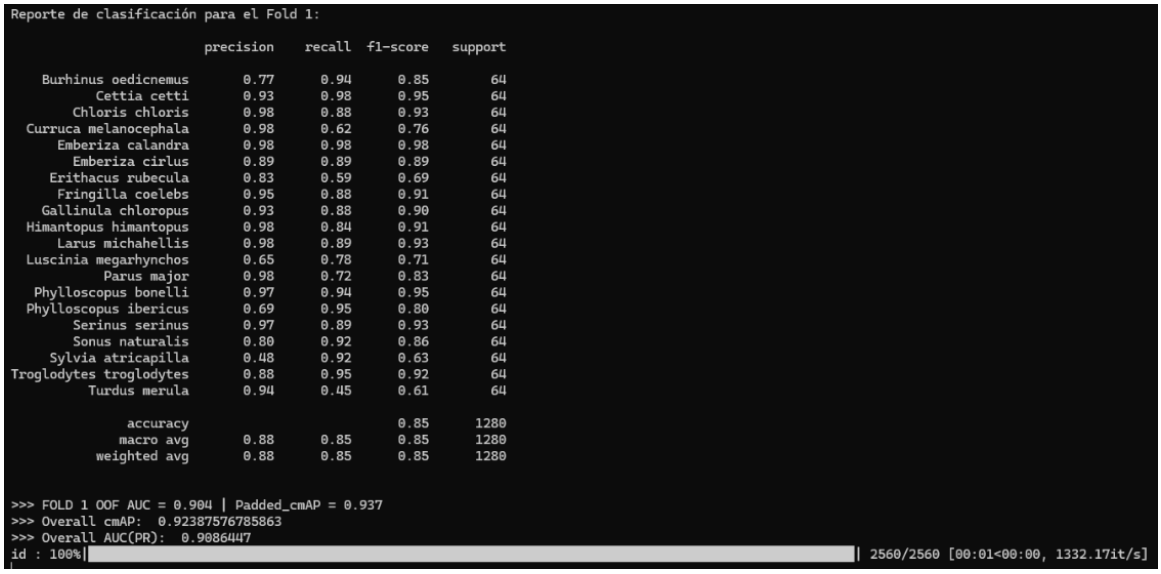
Además, se actualiza explícitamente class_names y se utilizan una serie de mapeos por medio de name2label y label2name que se aplican a los datos de entrenamiento y validación después del filtrado. Para detectar desequilibrios en el dataset se analiza su distribución a través del comando value_counts() en el dataframe de entrenamiento.

El nuevo archivo principal clasification_module.py realiza tras la adaptación múltiples verificaciones tales como la verificación de si está vacío el df_train después del filtrado, eliminación de filas con valores NaN en la columna target, asignación de folds y verificación de que cada fold tenga suficientes muestras. Respecto a los callbacks, la adaptación de Virtual Profiler cuenta con ModelCheckpoint donde cada fold el modelo es guardado con extensión .h5

en lugar de la extensión .keras, además se utiliza EarlyStopping cuando val_auc no mejora durante el entrenamiento durante 5 épocas consecutivas. Tanto en la versión original como en la versión adaptada se incluye un método denominado upsample_data a partir del cual se equilibran las clases, sin embargo, la versión adaptada incluye verificaciones para garantizar que los datos procesados estén en el formato correcto antes de continuar. A nivel de depuración la versión modificada de Virtual Profiler presenta una depuración mucho más extensa y exhaustiva que la versión original.

3.3.3 Investigación de arquitecturas de peso ligero,

Se han implementado tres modelos bajo la adaptación de un modelo existente. Estos son EfficientNetB0 con una extracción de características donde se utilizan los espectrogramas de mel, EfficientNetB0 con una extracción de características donde se utilizan los coeficientes cepstrales de mel (MFCCs) y MobileNet con espectrogramas simples. En todos ellos se ha utilizado la validación cruzada k-folds. A continuación, se incluyen los resultados que devuelve el modelo EfficientNetB0 por medio de los espectrogramas de mel:



Reporte de clasificación para el Fold 1:

	precision	recall	f1-score	support
Burhinus oedicnemus	0.77	0.94	0.85	64
Cettia cetti	0.93	0.98	0.95	64
Chloris chloris	0.98	0.88	0.93	64
Curruca melanocephala	0.98	0.62	0.76	64
Emberiza calandra	0.98	0.98	0.98	64
Emberiza cirlus	0.89	0.89	0.89	64
Erithacus rubecula	0.83	0.59	0.69	64
Fringilla coelebs	0.95	0.88	0.91	64
Gallinula chloropus	0.93	0.88	0.90	64
Himantopus himantopus	0.98	0.84	0.91	64
Larus michahellis	0.98	0.89	0.93	64
Luscinia megarhynchos	0.65	0.78	0.71	64
Parus major	0.98	0.72	0.83	64
Phylloscopus bonelli	0.97	0.94	0.95	64
Phylloscopus ibericus	0.69	0.95	0.80	64
Serinus serinus	0.97	0.89	0.93	64
Sonus naturalis	0.80	0.92	0.86	64
Sylvia atricapilla	0.48	0.92	0.63	64
Troglodytes troglodytes	0.88	0.95	0.92	64
Turdus merula	0.94	0.45	0.61	64
accuracy			0.85	1280
macro avg	0.88	0.85	0.85	1280
weighted avg	0.88	0.85	0.85	1280

>>> FOLD 1 OOF AUC = 0.904 | Padded_cmAP = 0.937
>>> Overall cmAP: 0.923875767858 63
>>> Overall AUC(PR): 0.9086447
id : 100%| 2560/2560 [00:01<00:00, 1332.17it/s]

Fig.4. Resultados EfficientNetB0 por MEL

El modelo ha convergido bastante rápido mostrando unos resultados aceptables con tan sólo 10 épocas.

A continuación, se presentarán los resultados del modelo EfficientNetB0 por medio de los coeficientes cepstrales de mel (MFCCs):

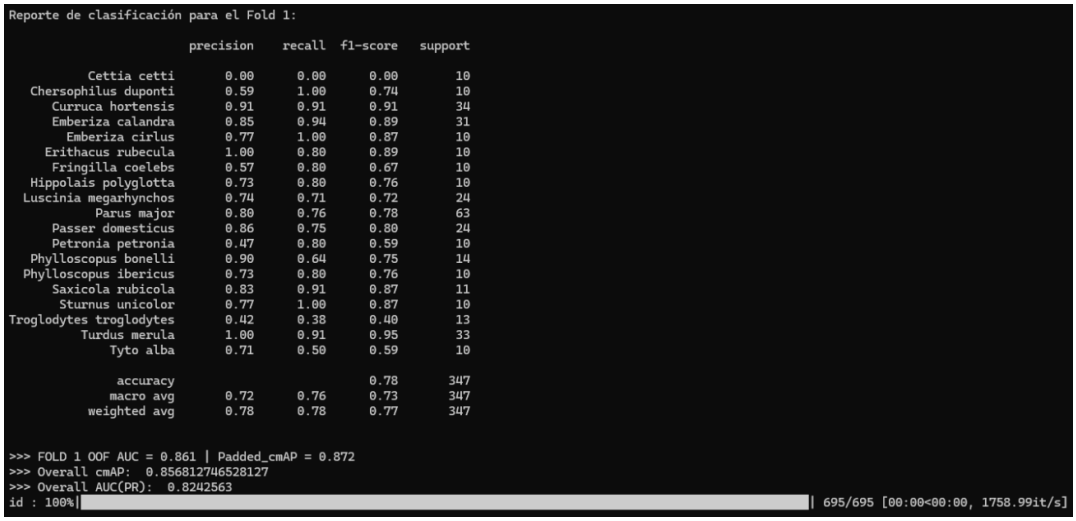


Fig.5 Resultados EfficientNetB0 por coeficientes cepstrales de MEL (MFCCs)

En esta ocasión el modelo muestra un accuracy ligeramente inferior a la extracción de características realizada mediante los espectrogramas de mel debido a que los coeficientes cepstrales de mel (MFCCs) son una característica derivada directamente de los espectrogramas de mel. Es por ello por lo que se está realizando una extracción de características más ligera lo que conduce a unos resultados más pobres. Por último, se tiene el modelo MobileNet con una extracción de características por medio del espectrograma simple.

A continuación, se muestran los resultados de dicho entrenamiento:

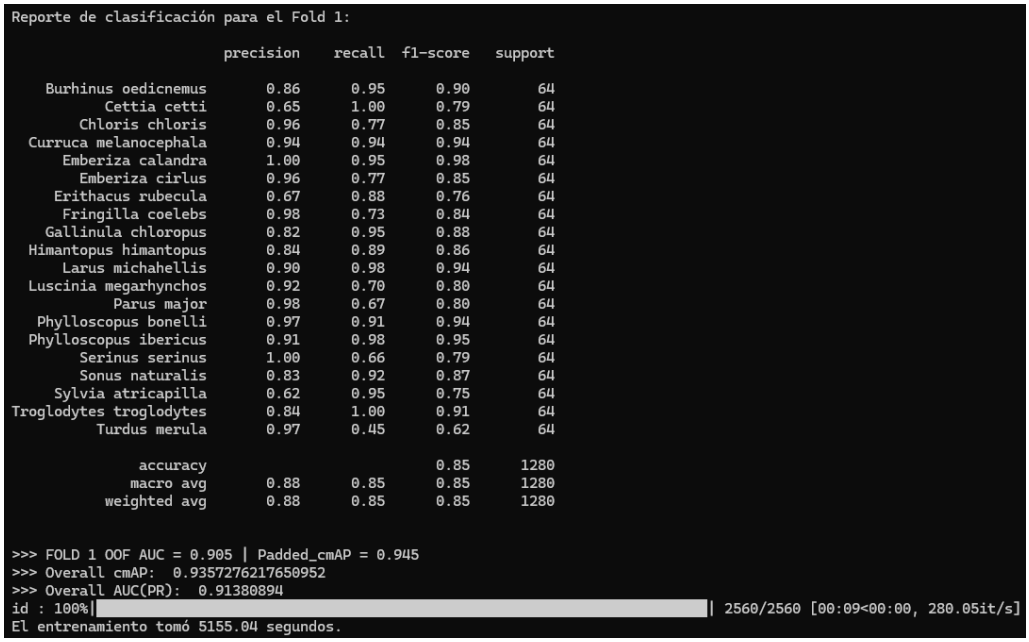


Fig.6 Resultados MobileNet por espectrograma simple

Como se puede observar el accuracy de la extracción de características sustentada en los espectrogramas simples vuelve a aumentar hasta la misma cantidad que los espectrogramas de mel. Es por ello por lo que a nivel de métricas no se observa una mejora sustancial entre el uso de espectrogramas simples o espectrogramas de mel en la realización de la extracción de características. Una vez analizados los resultados de los tres modelos se va a mostrar un espectrograma centrado en la captación de frecuencias altas

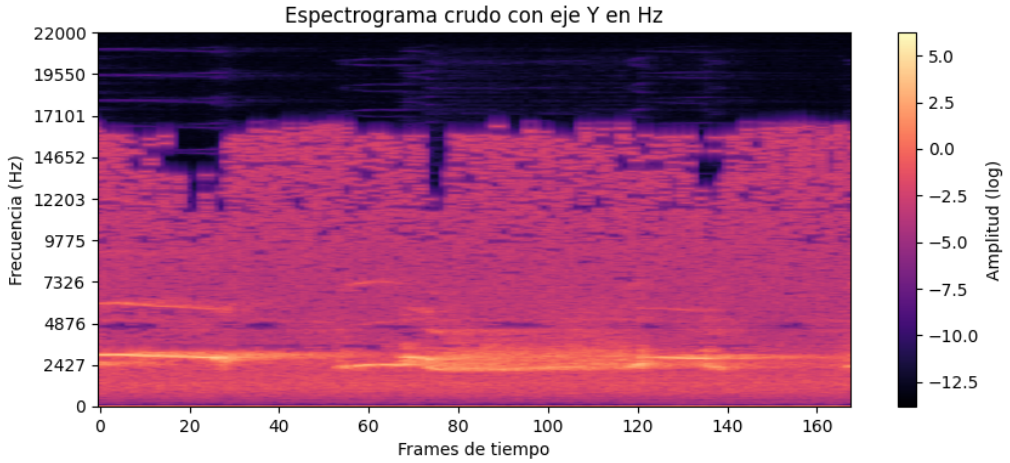


Fig.7 Espectrograma crudo

En el eje X se tienen frames de tiempo, donde se muestra el tiempo dividido en segmentos, este eje se extiende desde 0 hasta aproximadamente 160 frames. El color indica la amplitud de las frecuencias en decibelios (dB) en escala logarítmica, donde los colores representan diferentes niveles de amplitud. Respecto a su interpretación, las zonas oscuras indican niveles bajos de energía y las barras y líneas horizontales indican ciertas notas musicales, tonos de llamadas o cualquier tipo de señal que tenga una frecuencia constante. El rango de frecuencias abordado es el rango que utiliza el oído humano para detectar sonidos y se puede intuir que no existe energía en las frecuencias altas (más allá de 17.000 Hz), sin embargo, las frecuencias se concentran en torno a los 2500 Hz dando como resultado ciertas formas y tonos musicales.

3.3.4 Entrenamiento y comparativa de resultados

Por ello se decidió iniciar la búsqueda de técnicas novedosas en el procesamiento de audio para la clasificación de cantos de aves donde lo que se priorizaría sería la mejora del rendimiento en términos de complejidad computacional y complejidad temporal frente al mantenimiento de las métricas de precisión como pueden ser el accuracy. En esta búsqueda de la innovación se retornó de nuevo a los desarrollos partiendo de cero sin usar otros modelos anteriores. Por una parte se utilizaron tres datasets distintos (CLO, XenoCanto y Birdclef). Con la idea de construir un modelo capaz de clasificar aves utilizando menos recursos se diseñó e implementó una red neuronal convolucional CNN 1D junto con un encoder cuyos resultados fueron satisfactorios en el sentido de que se logró una reducción de tiempo y FLOPS sin perder demasiada accuracy. En contrapartida se decidió utilizar EfficientNetB0 para simular las técnicas actuales que se están desarrollando en el estado del arte, dicho modelo logró mejores métricas, pero con unos costes temporales y computacionales mayores. La clave de la innovación ha sido la utilización del audio en crudo para alimentar la CNN1D junto con el encoder en lugar de utilizar los espectrogramas que suelen usarse para alimentar a la EfficientNetB0. A continuación se

presenta una tabla comparativa con los resultados obtenidos que justifican la redacción del paper:

Algoritmo	Dataset	Accuracy	Top5Acc	Parámetros	Tiempo 1 muestra (s)	MAC 1 muestra (sin spec)
cnn1+Encoder	CLO	79.7421	0.955801	4.907.451	0.0203669	53.287.068
cnn1+Encoder	Xenocanto	80.5380	0.949346	4.936.299	0.0218868	53.315.868
cnn1+Encoder	birdclef	78.6739	0.925652	4.959.137	0.0218724	53.338.668
efficientnetb0	CLO	89.6994	0.974552	4.123.543	0.0512480	76.798.464
efficientnetb0	Xenocanto	92.5869	0.973478	4.172.221	0.0476222	76.847.104
efficientnetb0	birdclef	66.8508	0.881215	4.062.055	0.0670516	76.737.024

Tabla.1. Comparación de resultados

Paralelamente y siguiendo la idea de innovar en base a las prestaciones del modelo seleccionado, se desarrolló una red neuronal convolucional CNN1D que utilizaba coeficientes cepstrales de mel para la extracción de características. A través de este modelo se permiten clasificar los cantos de las aves con una complejidad reducida frente a los modelos desarrollados con anterioridad. Se necesitaron 77 ms para completar un step y las métricas resultantes son las que aparecen en la siguiente imagen:

```
80/80 ━━━━━━━━━━━━━━━━━━━━ 6s 77ms/step - accuracy: 0.9606 - loss: 0.1223 - val_accuracy: 0.7336 - val_loss: 2.1959 - learning_rate: 1.2500e-04
Training time: 84.75800728797913 seconds
35/35 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step
                       precision    recall  f1-score   support

    Burhinus oedicnemus       0.88      0.67      0.76        55
          Cettia cetti        0.63      0.76      0.69        55
        Chloris chloris       0.71      0.75      0.73        55
  Curruca melanocephala       0.85      0.82      0.83        55
     Emberiza calandra        0.92      0.87      0.90        55
       Emberiza cirlus        0.80      0.82      0.81        55
    Erithacus rubecula        0.67      0.73      0.70        55
     Fringilla coelebs        0.80      0.75      0.77        55
   Gallinula chloropus        0.71      0.71      0.71        55
  Himantopus himantopus       0.65      0.76      0.70        55
     Larus michahellis        0.74      0.71      0.72        55
 Luscinia megarhynchos        0.70      0.69      0.70        55
        Parus major           0.74      0.56      0.64        55
  Phylloscopus bonelli        0.69      0.45      0.55        55
 Phylloscopus ibericus        0.73      0.80      0.77        55
       Serinus serinus        0.85      0.95      0.90        55
       Sonus naturalis        0.84      0.95      0.89        55
    Sylvia atricapilla        0.70      0.73      0.71        55
Troglodytes troglodytes       0.76      0.82      0.79        55
        Turdus merula         0.75      0.78      0.77        55

             accuracy                             0.75      1100
            macro avg         0.76      0.75      0.75      1100
         weighted avg         0.76      0.75      0.75      1100
```

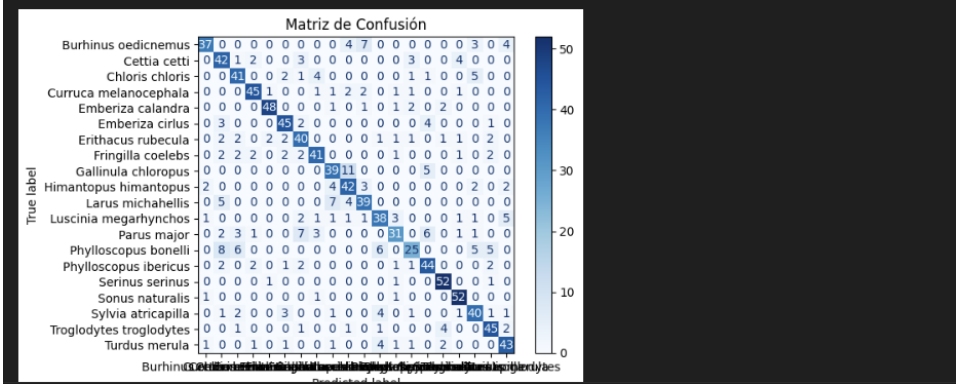


Tabla.2. Comparación de resultados 2

De manera similar y paralela al desarrollo descrito, se desarrolló una red neuronal convolucional CNN 1D junto a un encoder con capas de atención personalizables donde la extracción de características fuese la utilización directa de los audios en crudo para seguir con la línea de innovación propuesta anteriormente. Los resultados en términos de accuracy fueron ligeramente inferiores a los de la prueba anterior logrando un 74% frente a su sólido 79% para el dataset de XenoCanto. El estudio realizado para la clasificación de aves se sustenta en un cuatro datasets provenientes de XenoCanto y Ebird, y otras fuentes. Después de toda la gran cantidad de modelos construidos y la amplia variedad de extracción de características realizada, nos basaremos en el modelo CNN1D junto con el encoder construido donde en la tabla 1 se muestran sus resultados comparativos. El resto de los modelos y pruebas realizadas sirven para mostrar como el estado del arte actual nos confirma la existencia de una gran variedad de técnicas diferentes a través de las cuales clasificar cantos de aves de manera óptima, pero quizás sin priorizar la eficiencia frente a las métricas resultantes.

4 Redes Neuronales Convolucionales para el procesamiento de mapas de sensibilidad de la vida silvestre

En esta sección se documenta toda la información previa al desarrollo de los distintos modelos relacionados con las imágenes de las aves. Por otra parte, se define la arquitectura y funcionamiento del módulo encargado de la monitorización visual de aves, así como el proceso de desarrollo seguido.

4.1 Estado del arte

En esta sección se procede a hacer un repaso sobre el estado del arte relacionado con la monitorización visual de las aves. Se hará énfasis tanto en la parte de detección de aves en imágenes (segmentación) como en la parte de clasificación de estas.

4.1.1 Clasificación de aves con sistemas de radares

El funcionamiento base de este método consiste en la instalación de radares meteorológicos y radares de vigilancia en aviones que vuelen a bajas altitudes (no más de 2 km) para detectar las aves. En el artículo [6] se explica que la razón de volar a baja altitud se debe a que la probabilidad de avistar aves por encima de los 2 km de altitud es realmente baja. Aquí se encuentra la principal desventaja de este método, y es que se necesitaría una gran cantidad de aviones volando a bajas altitudes para poder detectar aves, siendo un gasto económico enorme. Además, los aviones comerciales no vuelan a esas altitudes (únicamente al despegar y aterrizar), así que habría que usar aviones de carácter militar. Un artículo que explica más explícitamente el uso del radar para la clasificación de aves es [7].

Otro artículo en el que también se definen los radares para la clasificación de aves haciendo una explicación del funcionamiento del radar es [8]. El radar es una tecnología que se desarrolló en los años 30 del siglo pasado, y desde entonces se ha ido optimizando y actualizando. Su funcionamiento base consiste en emitir pulsos de ondas electromagnéticas en una dirección. Las ondas al llegar al objeto (aves en este caso) rebotan y se detectan con una antena. Dependiendo de distintas condiciones físicas del objeto en el que la onda ha rebotado (forma, tamaño, velocidad, etc.) la onda rebotada tendrá unas características u otras. Aunque con esta tecnología teóricamente pueda ser posible reconocer y clasificar aves, las restricciones para llevar a cabo los experimentos son demasiado exigentes.

4.1.2 Clasificación de aves con Redes Convolucionales

Las redes convolucionales (CNN) son un subconjunto de las redes neuronales artificiales (ANN). Este tipo de redes realizan una serie de operaciones (convoluciones) que resultan de gran utilidad para el procesamiento de imágenes, pues son capaces de procesar la información espacial que tiene cada uno de los píxeles. El artículo [9] hace una explicación detallada del funcionamiento de estas redes a nivel teórico, sin aplicarlo a las aves. Otros autores también han escrito artículos con conclusiones y métodos similares [10]. Otras operaciones que realizan estas redes son operaciones de pooling, explicadas en el artículo [11], y funciones de activación, definidas en el artículo [12].

La idea principal subyacente tras estas redes consiste en crear una tubería de procesamiento de capas convolucionales en serie, de forma que las salidas (llamadas características) de la capa n sean la entrada de la capa n+1. Esta tubería de procesamiento se encarga de hacer el proceso llamado extracción de características, que es el proceso en el cual se extraen características específicas de la imagen. Posteriormente esas características se hacen pasar por una red fully connected para clasificar la imagen. En la Figura 8 se describe un ejemplo de una red convolucional con arquitectura VGG16, la arquitectura CNN más sencilla.

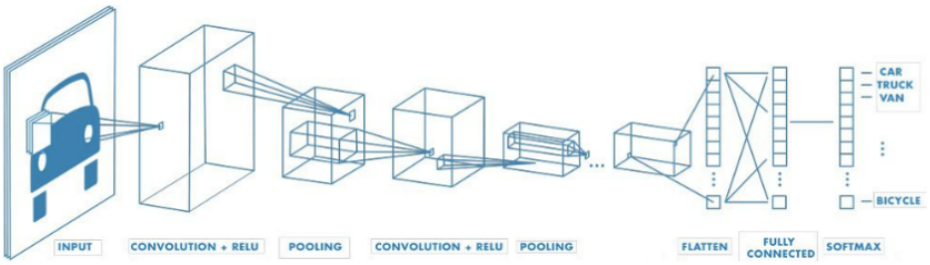


Fig.8 Ejemplo funcionamiento CNN

No obstante, a pesar de que las CNN son capaces de extraer la información espacial de cada píxel, presentan algunas destacables desventajas, como la alta capacidad de cómputo necesaria para entrenarlas y la gran cantidad de imágenes necesarias para poder generalizar el concepto objetivo. Con fin de paliar estos problemas se desarrollaron otros tipos de redes como las redes residuales (eliminan el problema de la degradación del gradiente y facilita el aprendizaje) [13] o redes densas (combinan características de distintas capas facilitando el aprendizaje [14]). Técnicas de aumentación de datos para obtener un mayor número de imágenes significativas que aporten información para generalizar el concepto también son técnicas muy comúnmente utilizadas en este ámbito [15]. En el artículo [16] se hace un resumen de las desventajas de las CNN, así como de posibles soluciones. La CNN que a fecha de diciembre de 2024 mejores resultados están ofreciendo en dominios de distinta índole, incluyendo la clasificación de aves, son las EfficientNet [17]. Es un tipo de redes en las que se ha conseguido disminuir drásticamente el número de parámetros necesarios a aprender, siendo muy eficientes. También resuelve problemas muy recurrentes en todo el campo del aprendizaje profundo, como el problema de la degradación, presente en todos los modelos definidos hasta la fecha. El artículo en el que se presentó por primera vez las EfficientNet es [18], definiendo los problemas que resuelve y las ventajas que presenta.

4.1.3 Clasificación de aves con Visual Transformers

Los Visual Transformers son una modificación de los transformers desarrollados en 2017 [19]. Los transformers inicialmente fueron desarrollados para ser utilizados en el ámbito de las series temporales. Las series temporales son un tipo de datos en el que el orden de los datos de la secuencia es relevante, y cada dato en esa secuencia se procesa uno tras otro (secuencialmente). Este tipo de redes fueron una gran revolución y son las más usadas en campos como el procesamiento de lenguaje natural o IA generativa. La definición de los transformers, la revolución de los mecanismos de Auto-Atención, su estructura e implementación se encuentran en el artículo [20]. La característica principal de las redes recurrentes es que la salida del dato n de una secuencia es la entrada de la red junto con el dato n+1. En la Figura 2 se ve una retro propagación de una neurona recurrente en el tiempo, es decir, la neurona es la misma en distintos instantes de tiempo. Las entradas son Xt−3 , Xt−2 , Xt−1 y Xt , mientras que las salidas son yt−3 , yt−2 , yt−1 , yt . Se puede apreciar que la salida de un instante cualquiera es la entrada del instante siguiente.

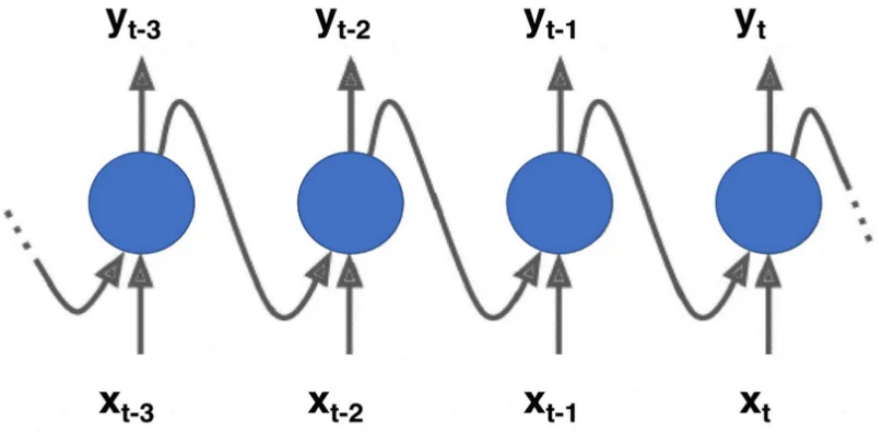


Fig.9 Ejemplo red neuronal recurrente procesando una secuencia

Los transformers se caracterizan por el uso de mecanismos de Auto-atención. Este mecanismo se basa en codificar el dato teniendo en cuenta 3 características del dato. En el ámbito del lenguaje natural, estas 3 características son: la palabra, el orden de palabras y el contexto. Esto se puede extrapolar a otros dominios donde los datos no sean palabras, como el dominio de las imágenes. Toda esta información está definida en el artículo [21].
La transformación que hay que realizar para que los transformers puedan clasificar imágenes es significativa, teniendo que dividir las imágenes en una serie de parches y hacer esas operaciones de Auto-atención en los píxeles de cada parche. El artículo en el que se mencionaron por primera vez los ViT y su implementación es [22].

4.1.4 Segmentación de aves con Redes Convolucionales

En el apartado anterior, se han definido las redes convolucionales para resolver problemas de clasificación, es decir, dar una imagen como entrada a la red para que esta la clasifique entre 1 de las n clases para las que fue entrenada. Sin embargo, estas redes ofrecen más funcionalidades en otros dominios, como la segmentación de objetos en imágenes. La segmentación de objetos en imágenes consiste en, dada una imagen con 1 o más objetos, identificar los n objetos existentes en dicha imagen.
Los objetos en la imagen pueden pertenecer a distintas clases, aunque en este caso serán solo aves. En la Figura 10 se puede ver un ejemplo de una imagen habiendo segmentado todas las aves. Los modelos de segmentación de imágenes son capaces tanto de segmentar los objetos en una imagen, como de clasificarlos. Otra opción es en un primer paso segmentar todos los objetos, y luego otro modelo es entrenado para clasificarlos. En este proyecto se seguirá la segunda aproximación.

Fig.10 Ejemplo de segmentación de aves en una imagen usando CNN

Para hacer segmentación de objetos en imágenes, el modelo más utilizado y el que mejores resultados presenta es el modelo YOLO, (‘You Only Look Once’ en inglés) que está basado en CNN. En el artículo [16] se hace uso del modelo YOLO para la segmentación de aves en imágenes.

4.1.5 Segmentación de aves por colores

Con fin de segmentar los objetos contenidos en una imagen, otro método extensamente utilizado antes de la llegada de las redes neuronales fue la segmentación por colores. El procedimiento de esta técnica empieza por eliminar el fondo de la imagen (se puede interpretar como ruido). Los colores de los bordes de la imagen se escanean y se hace un ranking según la frecuencia de aparición de cada color en un histograma. Se establece un umbral y una serie de heurísticas para definir qué colores son fondo de la imagen y cuáles no. A continuación, se recorren todos los píxeles de la imagen comparando su color con la información ofrecida por el histograma, siendo considerado fondo u objeto en cada caso.

En caso de que la imagen esté en blanco y negro, habrá un solo histograma. Si la imagen tiene 3 canales de color (RGB), tendrá un histograma por cada canal. El paso de seleccionar el umbral a partir del cual unos colores se consideran fondo y otros no, es un proceso iterativo de prueba y error, que depende del dominio y características de las imágenes. En el artículo [23] se detalla el proceso seguido para hacer la segmentación por colores explicada anteriormente.

Aunque la segmentación por colores es un método que requiere poca capacidad de cómputo, es una técnica que presenta numerosas desventajas. Entre ellas cabe destacar que, dependiendo de las características de la imagen, puede que el fondo no sea muy diferenciable de los objetos. En el caso específico de las aves, en caso de que el fondo de la imagen no presente un color uniforme (si hay nubes, distinta luminosidad u otros factores que afecten a la diferencia de colores entre las aves y el fondo) la tasa de acierto de este modelo para segmentar aves se reduce drásticamente. Como las redes neuronales resuelven este problema aprendiendo este tipo de peculiaridades (las CNN aprenden a reconocer los objetos sin importar los colores del fondo), las redes neuronales han sustituido a esta técnica en los últimos años.

4.2 Arquitectura

4.2.1 Datos

La cámara usada para la adquisición de datos en el proyecto IA4BIRDS es AXIS Q6225-LE PTZ Camera [18]. Con dicha cámara se han realizado vídeos del entorno bajo distintas condiciones meteorológicas para capturar imágenes de aves en distintas condiciones, y así evitar sesgos y bias que dificulten el entrenamiento de los modelos.
Los fotogramas de los vídeos capturados tienen una resolución de 1920 x 1080 píxeles, con 3 canales de color (RGB), de forma que cada fotograma de un vídeo cuenta con un total de

*1920 * 1080 * 3 = 6.220.800* valores.

Con estos vídeos, se han entrenado 2 modelos distintos. El primero de ellos es un modelo YOLOv8, basado en redes convolucionales que se encarga de identificar o segmentar las aves en el vídeo. El segundo el Yolov11, que introduce mejoras en eficiencia, generalización y entrenamiento dinámico.
Una vez hecho eso, con esas imágenes de aves identificadas en los vídeos, se entrenaron modelos que clasifican cada una de esas aves.

4.2.2 Calibración de la cámara

La cámara usada para la adquisición de datos en el proyecto IA4BIRDS es AXIS Q6225-LE PTZ Camera [18].

La resolución del sensor de la cámara es 1920x1080 píxeles. El tamaño de cada píxel es 3.75 µm x 3.75 µm. Por tanto:

- El tamaño horizontal del sensor de la cámara es 1920 px * 3.75µm = 7.2 mm
- El tamaño vertical del sensor de la cámara es 1080 px * 3.75µm = 4.05 mm

Según la documentación oficial de la cámara, la distancia focal mínima y máxima es 6.91 mm y 214.34 mm respectivamente. Los siguientes cálculos se han realizado basado en los datos anteriores.

La cámara no ofrece directamente la distancia focal usada en la grabación de cada fotograma, pero sí ofrece el aumento de zoom, el cual tiene una relación directa con la distancia focal. Dicha relación es:

- distancia focal usada (mm) = aumento de zoom * distancia focal mínima (mm)

En este caso, la fórmula anterior quedaría:

- distancia focal usada (mm) = aumento de zoom * 6.91 mm

La API de la cámara no proporciona directamente funcionalidad para controlar el aumento de zoom empleado en cada fotograma, pero sí provee un parámetro llamado 'paso de zoom' , contenido en el intervalo [1,7986] que tiene una relación directa con el aumento de zoom. Tanto la información de paso de zoom como aumento de zoom usado en cada fotograma es ofrecida en el mismo. Existe una opción para aumentar el máximo paso de zoom de 7986 a 9999. No obstante, como no se han detectado diferencias significativas, no se ha activado dicha opción.

Debido a que la relación que hay entre la distancia focal y el HFOV ('Horizontal Field of View' del inglés) y el VFOV ('Vertical Field of View' del inglés) es desconocida, se ha calculado este mapeo experimentalmente. Dicho mapeo se muestra en la tabla 3.

Paso de Zoom	HFOV	VFOV	Distancia Focal
1 (min)	60.28°	34.7°	6.91 mm
500	22°	12.48°	17.275 mm
800	15.28°	8.8°	23.494 mm
1000	12.48°	7.12°	27.64 mm
1200	10.48°	5.84°	31.786 mm
1400	8.68°	5°	35.932 mm
1600	7.44°	4.24°	40.078 mm
1800	6.2°	3.64°	44.224 mm
2000	5.34°	3°	48.37 mm
2500	3.92°	2.28°	58.735 mm
3000	3.28°	1.94°	69.1 mm
4000	2.46°	1.46°	89.83 mm
5000	1.98°	1.14°	110.56 mm
6000	1.64°	1.02°	131.29 mm
7986 (max)	1.28°	0.78°	172.75 mm

Tabla.3. Relación entre Zoom, Ángulo de campo de visión y distancia focal

Tener en cuenta que los valores de la Tabla 1 han sido obtenidos experimentalmente, por tanto están sujetos a cierto error. Se ha creado un modelo de interpolación basado en splines cúbicos para aproximar aquellos valores del paso de zoom no representados en la Tabla 1 (ej. paso de zoom = 6321).

4.2.3 Estudio Zoom óptimo

Se realizó un estudio cuantitativo acerca de que zoom es el óptimo en términos de cantidad y tamaño de las aves detectadas. Además, de manera previa se realizó este mismo estudio de las distancias, pero sin especificar un zoom concreto con el que centrar los esfuerzos. Es decir, se realizó un estudio para estimar la distancia a la que se encuentran las aves según su posible especie utilizando datos de todos los zooms. Los resultados del primer estudio (el de los zooms) mostraron que los zooms óptimos son el 17 y 19 en términos de cantidad de aves detectadas y tamaño de las aves detectadas. Por esa razón se centra la atención del presente estudio en el cálculo de las distancias de las aves con un zoom de 19.

A continuación, se crea un único archivo las anchuras de las aves para su posterior cálculo de las distancias estimadas. En primer lugar, se define el directorio de entrada y el directorio de salida. Se define el nombre del archivo de salida y se recorren todos los archivos .txt del directorio de entrada, se abren dichos archivos en modo lectura y codificación UTF-8 para recorrer cada línea de cada documento y extraer su tercera columna correspondiente a la anchura del ave para unificarlos en un único archivo .txt.

Lo siguiente que se hace es contar la cantidad de aves detectadas aunque sean repetidas en un vídeo de 4 horas y media. Se define una función que incrementa un contador por cada línea del archivo y devuelve dicho contador con el objetivo de llevar un conteo del número de líneas del archivo. Cada ave aunque se repita representa una línea. Se define la ruta del archivo para leerlo, se llama a la función recién definida y se muestra el resultado devuelto por la función.

```
FUNCION contar_aves(archivo)
    contador <- 0
    ABRIR archivo EN MODO LECTURA
        PARA CADA linea EN archivo HACER
            contador++
        FIN PARA
    CERRAR archivo
    RETORNAR contador
FIN FUNCION
ruta_archivo <- "anchurasAves.txt"
numero_filas <- contar_aves(ruta_archivo)
IMPRIMIR("El número de filas en el archivo es:", numero_filas)
```

CREACIÓN DE LOS .TXT CON LAS DISTANCIAS MÍNIMAS Y MÁXIMAS

En primera instancia, se importa la clase Calibrator, la cual contiene funciones geoespaciales para calcular distancias, azimuts absolutos, colatitudes absolutas, etc. Se define la función calcular_distancias, en dicha función se instancia Calibrator con la latitud y la colatitud del edificio Air Institute ya que es la zona donde está ubicada la cámara. Se convierte la longitud real del ave de cm a metros tanto la longitud mínima como la máxima, se establece el nivel de zoom a 19 y se calcula el focal_length multiplicando el nivel de zoom por la constante 6.91. Además, se abre el archivo de entrada en modo lectura y se cargan todas las líneas, posteriormente se abre el archivo de salida en modo escritura para recorrer cada línea y obtener la anchura del ave detectada. Posterior a esta obtención del número de píxeles se utiliza la función compute_distance_1D; el propósito principal de esta función es determinar cuán lejos está un objeto de la cámara, basándose en:

- La longitud focal de la cámara (focal_length).
- El tamaño real del objeto (por ejemplo, la envergadura o el ancho del ave, en metros).
- El tamaño del objeto en la imagen, medido en píxeles (n_pixels).

A continuación, se le asigna a cada línea (ave) un ID único y por último se escribe de manera estructurada en el archivo de salida. Nótese que existirá un archivo de salida por cada especie cuyo título será la propia especie y cuyo contenido será estructurado de forma que se incluya el número de pixeles, la especie, el id de la imagen, la distancia mínima y la distancia máxima. Por último, se llama a la función previamente definida calcular distancias con las dimensiones reales del ave y el nombre de cada especie. Esta función será llamada tantas veces como especies existan, cargando sus respectivos argumentos.

```
IMPORTAR Calibrator DESDE Calibrator
```

```
FUNCION calcular_distancias(real_length_min, real_length_max, especie)
    calibrator <- NUEVO Calibrator(camera_latitude=40.97, camera_longitude=-5.63)
    real_length_min <- real_length_min / 100
    real_length_max <- real_length_max / 100
    zoom_level <- 19
    focal_length <- zoom_level * 6.91
    archivo_entrada <- "anchurasAves.txt"
    archivo_salida <- CONCATENAR(especie, ".txt")
    SI EXISTE_ARCHIVO(archivo_entrada) ENTONCES
        ABRIR archivo_entrada EN MODO LECTURA
            lineas <- LEER_TODO(archivo_entrada)
        CERRAR archivo_entrada
        ABRIR archivo_salida EN MODO ESCRITURA
            PARA CADA idx, linea EN ENUMERAR(lineas) HACER
                n_pixels <- CONVERTIR(ELIMINAR_ESPACIOS(linea), REAL)
                distancia_min <- calibrator.compute_distance_1D(focal_length,
real_length_min, n_pixels)
                distancia_max <- calibrator.compute_distance_1D(focal_length,
real_length_max, n_pixels)
                id_imagen <- CONCATENAR(zoom_level, "-", idx + 1)
                ESCRIBIR(file, CONCATENAR(n_pixels, " ", especie, " ", id_imagen, " ",
distancia_min, " ", distancia_max))
            FIN PARA
        CERRAR archivo_salida
    FIN SI
FIN FUNCION
// Ejemplo de uso de la función
calcular_distancias(40, 42, "paloma_torcaz")
```

CREACIÓN DEL CSV

Lo primero que se hace es definir la función crear_dataset, dicha función combina múltiples archivos .txt en un único archivo CSV, intentando con codificaciones alternativas si es necesario. Se carga una lista con el nombre de los archivos .txt existentes en el directorio enviado como argumento, dichos nombres corresponden al nombre de las especies. Se define la ruta del archivo CSV resultante como el directorio seguido del nombre del archivo CSV. Se abre el archivo CSV en modo escritura y con codificación UTF-8, se instancia un csv.writer a partir del archivo recién abierto, se escribe a través del csv.writer el nombre de las columnas que en este caso son N_Pixels, Especie, ID_Imagen, Distancia_Min y Distancia_Max. Se recorre cada archivo de los cargados en la lista y se intentan abrir en modo de lectura, se recorre cada línea del archivo leído para extraer su información y se escribe esa información en el archivo de salida mediante el

writer. En caso de obtener un UnicodeDecodeError, entonces se trata de abrir cada archivo de los cargados en la lista en modo lectura para extraer línea a línea y escribir esos datos en el archivo de salida a través del writer con codificación ISO-8859-1. Este traspaso de datos de los archivos .txt al archivo CSV es viable gracias a la estructuración con la que han sido almacenados en los archivos .txt. Por último, se define el directorio y se le envía como argumento a la función crear_dataset.

```
FUNCION crear_dataset(directorio)
    archivos_txt <- LISTA(ARCHIVOS '.txt' EN directorio)
    archivo_csv <- CONCATENAR(directorio, 'dataset_especies.csv')

    ABRIR archivo_csv EN MODO ESCRITURA CON FORMATO "utf-8"
        ESCRIBIR_LINEA(archivo_csv, ['N_Pixels', 'Especie', 'ID_Imagen',
'Distancia_Min', 'Distancia_Max'])
        PARA CADA archivo EN archivos_txt HACER
            ruta_completa <- CONCATENAR(directorio, archivo)

            INTENTAR
                ABRIR ruta_completa EN MODO LECTURA CON FORMATO 'utf-8'
                    PARA CADA linea EN ruta_completa HACER
                        datos <- LISTA(ELIMINAR_ESPACIOS(linea))
                        ESCRIBIR_LINEA(archivo_csv, datos)
                    FIN PARA
                CERRAR ruta_completa
            CAPTURAR UnicodeDecodeError
                ABRIR ruta_completa EN MODO LECTURA CON FORMATO 'ISO-8859-1'
                    PARA CADA linea EN ruta_completa HACER
                        datos <- LISTA(ELIMINAR_ESPACIOS(linea))
                        ESCRIBIR_LINEA(archivo_csv, datos)
                    FIN PARA
                CERRAR ruta_completa
            FIN INTENTAR
        FIN PARA
    CERRAR archivo_csv
FUNCION
// Ejemplo de uso
directorio <- 'Especies_Z19'
crear_dataset(directorio)
```

ELIMINACIÓN DE VALORES ATÍPICOS

Para evitar sesgar las gráficas y obtener conclusiones erróneas utilizaremos el rango intercuartílico (Q1 - Q3). De esta manera no se tienen en cuenta valores extremos que pueden alterar las distancias a las que se encuentren las aves. En primer lugar, se define la función remover_outliters la cual almacena el primer cuartil en Q1, el tercer cuartil en Q3 y el rango intercuartílico en IQR. Se calcula el límite inferior y superior como Q1 - 1.5 * IQR y Q3 + 1.5 * IQR respectivamente. Posteriormente se define un data frame vacío denominado data_clean y se recorre cada especie única, se obtienen los datos de esa especie en concreto, se eliminan los outliers por medio de la función recién definida tanto de la distancia mínima como de la distancia máxima y por último se concatenan los resultados en el data frame data_clean. A continuación, se adjuntará el código utilizado para ello:

```
FUNCION remover_outliers(dataframe, columna)

    Q1 <- CUANTIL(dataframe[column], 0.25)
    Q3 <- CUANTIL(dataframe[column], 0.75)
    IQR <- Q3 - Q1
    lower_bound <- Q1 - 1.5 * IQR
    upper_bound <- Q3 + 1.5 * IQR
    RETORNAR FILTRAR(df, df[columna] >= lower_bound Y df[columna] <= upper_bound)

FIN FUNCION
// Aplicar la eliminación de outliers para cada tipo de distancia
data_clean <- pandas.DataFrame()  // Crear un DataFrame vacío para los datos limpios
PARA CADA specie EN VALORES_UNICOS(data['Especie']) HACER

    specie_data <- FILTRAR(data, data['Especie'] == specie)
    specie_data_clean_min <- remover_outliers(specie_data, 'Distancia_Min')
    specie_data_clean_max <- remover_outliers(specie_data_clean_min, 'Distancia_Max')
    data_clean <- pandas.concat([data_clean, specie_data_clean_max], axis=0)
```

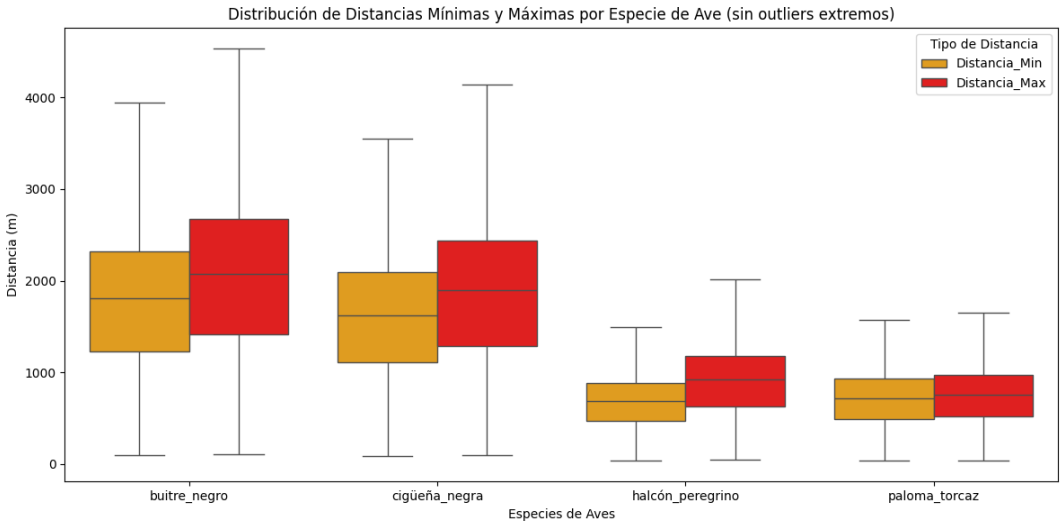


Fig.11 Distribución distancias mínimas y máximas por especie de Ave

GRÁFICAS E INTERPRETACIÓN

La gráfica muestra los boxplots que representan la distribución de las distancias mínimas y máximas observadas en cuatro especies diferentes de aves. Cada especie tiene dos representaciones: una para la Distancia_Min (en amarillo) y la otra para la Distancia_Max (en rojo). Los elementos del boxplot son:

La mediana (línea central en cada boxplot), la cual indica el valor medio de la distribución de distancias para cada medición, separando el 50% superior de datos del 50% inferior.
Caja (IQR): El rango intercuartil desde el primer cuartil (Q1, 25%) hasta el tercer cuartil (Q3, 75%), muestra la dispersión central de los datos. Cuanto más compacta es la caja, más agrupados están los datos en torno a la mediana.
Bigotes: Extienden desde el borde de la caja hasta el valor máximo y mínimo dentro del 1.5 * IQR desde el Q1 y Q3, respectivamente. Los puntos fuera de estos bigotes se consideran valores atípicos.
Interpretación: No existen valores atípicos debido a que se suprimieron aquellos valores que quedaban fuera del rango intercuartílico con el objetivo de obtener unas medidas más cercanas a la realidad y a la poca cantidad de valores presentes fuera del rango.

El buitre negro posee una mediana en torno a los 1800 metros en la distancia mínima y en torno a los 2100 metros en la distancia máxima, además la distancia a la que se suele encontrar es entre 1500 metros y 2500 metros.

La cigüeña negra posee una mediana en torno a los 1600 metros en la distancia mínima y en torno a los 2000 metros en la distancia máxima, además la distancia a la que se suele encontrar es entre 1200 metros y 2300 metros.

El halcón peregrino posee una mediana en torno a los 600 metros en la distancia mínima y en torno a los 800 metros en la distancia máxima, además la distancia a la que se suele encontrar es entre 500 metros y 1100 metros.

La paloma torcaz posee una mediana en torno a los 500 metros en la distancia mínima y en torno a los 600 metros en la distancia máxima, además la distancia a la que se suele encontrar es entre 300 y 800 metros.

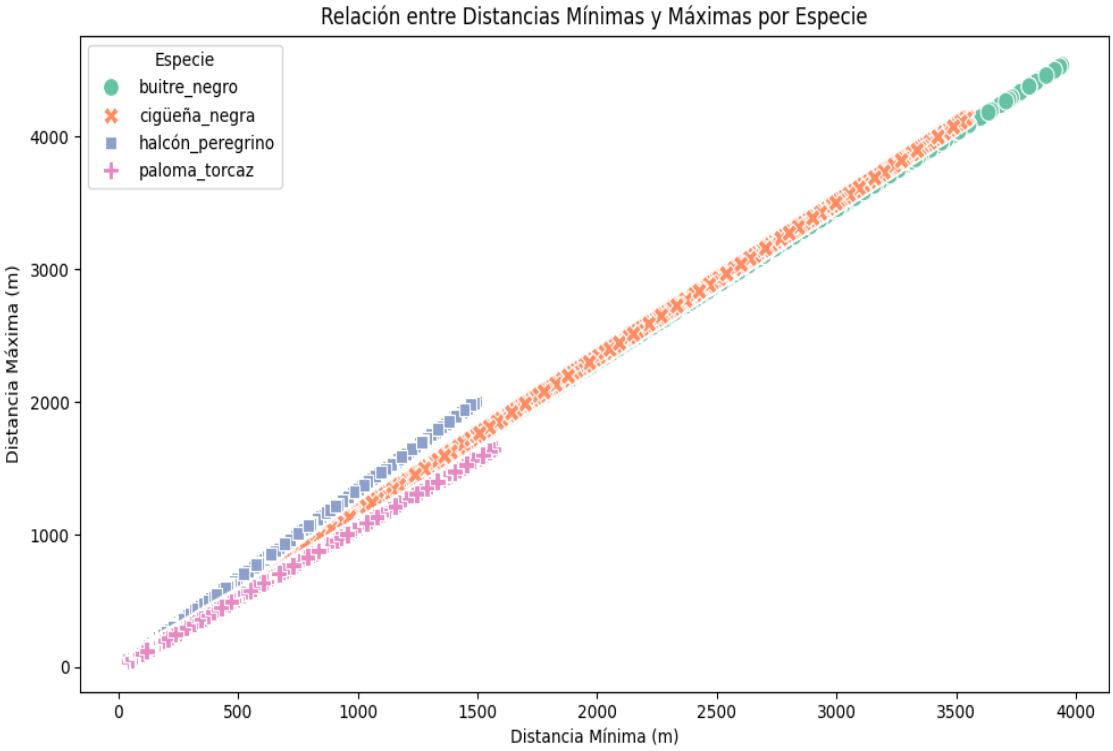


Fig.12 Relación entre distancias mínimas y máximas por especie

La gráfica muestra un gráfico de dispersión donde se comparan las distancias mínimas y máximas registradas por cada especie de ave. Los puntos están codificados por colores y símbolos que representan diferentes especies:

- Círculos verdes: Buitre negro
- Cruces naranjas: Cigüeña negra
- Cuadrados morados: Halcón peregrino
- Cruces rosas: Paloma torcaz

Existe cierta correlación lineal aparente, para todas las especies, hay una correlación lineal clara entre las distancias mínimas y máximas, lo que indica que las mayores distancias mínimas generalmente se asocian con mayores distancias máximas dentro de la misma observación. Cada especie muestra una tendencia lineal, pero las pendientes y las intercepciones con el eje (y) varían, lo que sugiere diferencias en cómo cada especie gestiona sus rangos de vuelo:

- Buitre negro y cigüeña negra: Muestran las mayores distancias, con los buitres negros alcanzando las mayores distancias máximas para cualquier distancia mínima dada.
- Halcón peregrino y paloma torcaz: Tienen distancias más contenidas, con los halcones peregrinos mostrando menor variabilidad entre las distancias mínimas y máximas comparando con las cigüeñas y los buitres.

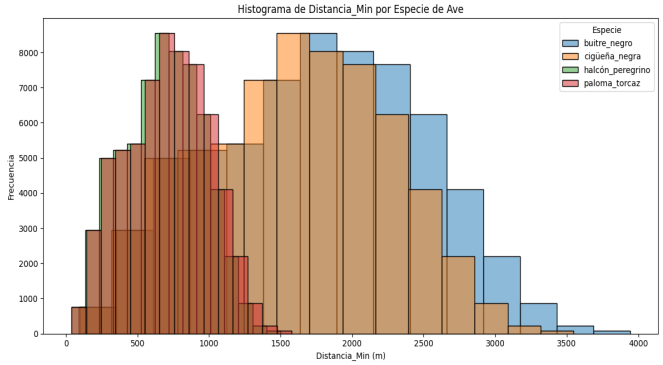


Fig.13 Histograma de distancia min por especie

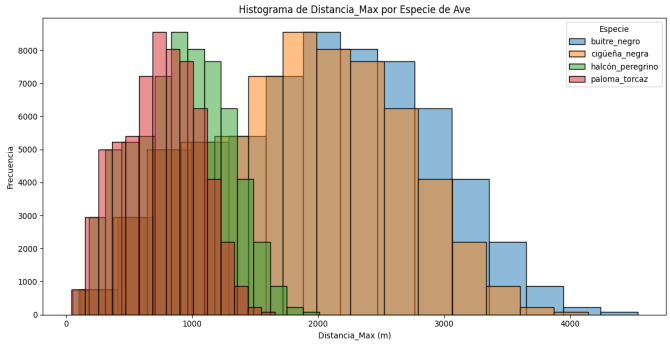


Fig.14 Histograma de distancia Máx por especie

Estas gráficas representan la distribución de las distancias mínimas y máximas observadas para cuatro especies de aves: buitre negro, cigüeña negra, halcón peregrino y paloma torcaz. Cada especie está representada por un color específico en dos gráficos separados, uno para la distancia mínima y otro para la distancia máxima.
En el gráfico de distancia mínima se puede observar al buitre negro con un color azul y un pico claro alrededor de los 1500 metros, mostrando una concentración alta de observaciones en esta distancia. Se puede observar a la cigüeña negra con un color naranja, su distribución es ligeramente más amplia con un pico cerca de los 1700 metros, indicando variabilidad, pero con una tendencia a concentrarse en esta distancia. Se puede observar al halcón peregrino con un color verde, presenta un pico agudo cerca de los 700 metros, reflejando una preferencia por distancias mínimas más cortas. Por último, se puede observar a la paloma torcaz con un color rojo, su pico está alrededor de los 750 metros, similar al halcón peregrino, pero probablemente con una distribución más estrecha.

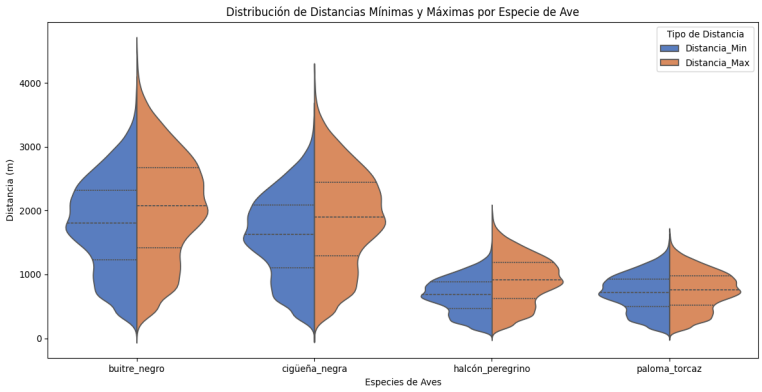


Fig.15 Distribución de distancias mínimas y máximas

La visualización muestra gráficos de violín para las distancias mínimas (en azul) y máximas (en naranja) de vuelo de cuatro especies de aves. Estos gráficos permiten comparar la dispersión y la distribución de los valores dentro de cada especie.

Los gráficos de violín presentan las siguientes características:

- Cuerpo del violín: La forma más ancha del violín indica una mayor densidad de datos, es decir, muchos puntos de datos se encuentran en esa región de distancia.
- Línea central (Mediana): La línea de puntos dentro de cada violín muestra la mediana de la distribución, proporcionando un punto de referencia para entender dónde se centran los datos.
- Anchura del violín: Refleja la distribución de los datos. Una base más ancha sugiere mayor variabilidad en las distancias registradas.

El buitre negro tanto para las distancias mínimas como para las máximas presenta una gran dispersión, indicando una variabilidad significativa en las distancias de vuelo. La cigüeña negra es similar al buitre negro, tiene una amplia dispersión en ambas distancias, con una densidad de datos que también se extiende a distancias mayores. El halcón peregrino presenta unos gráficos de violín relativamente estrechos, especialmente para las distancias máximas, indicando menos variabilidad y distancias más consistentes. y distancias más consistentes. La paloma torcaz presenta violines más estrechos, lo que sugiere la menor variabilidad entre las especies estudiadas y distancias consistentemente menores.

4.2.4 Estudio de las distancias y tamaños de las aves

El último parámetro para poder calcular la distancia a la que se encuentra el ave es la longitud en metros. Al no disponer de esta información de manera certera, se ha probado con las dimensiones de tres tipos de aves diferentes. Estas aves son buitre negro, halcón peregrino, paloma torcaz y cigüeña negra. Todas ellas son aves grandes debido a que las más pequeñas no serán identificadas por nuestra cámara.

Ave	Longitud (cm)	Envergadura (cm)	Velocidad (km/h)	Velocidad Media (km/h)
Buitre negro	100 - 115	265 - 290	40 - 50	45
Halcón peregrino	38 - 51	80 - 120	60 - 90	75
Paloma torcaz	40 - 42	75 - 80	60 - 70	65
Cigüeña negra	90 - 105	145 - 155	50 - 70	60

Tabla.4. Comparativa tamaño y velocidad aves

¿CÓMO CALCULAR LA DISTANCIA A LA QUE SE ENCUENTRA EL AVE?

Para calcular la distancia a la que se encuentra el ave es necesario conocer el Focal Lenght de la cámara, el número de píxeles de anchura y la longitud real en metros del ave. El cálculo se realiza por medio de la instanciación de la clase Calibrator, es por ello por lo que también resulta imprescindible conocer la longitud y latitud a la que se encuentra la cámara. Por medio de la siguiente ilustración se especifica lo explicado recientemente:

PROCESAR ARCHIVOS DE TEXTO CON LAS ETIQUETAS PARA SELECCIONAR SOLO AQUELLOS QUE TIENEN CIERTA CANTIDAD DE AVES DETECTADAS

El código propuesto tiene como objetivo principal optimizar la gestión y preparación de datos etiquetados que son usados en el entrenamiento de modelos de visión por computadora. Funciona mediante dos procesos clave: primero, escala las dimensiones normalizadas de los objetos detectados (anchura y altura) a valores en píxeles, adecuados a una resolución estándar de 1920x1080. Esta conversión es fundamental para adaptar los datos a formatos compatibles con algoritmos de aprendizaje automático que requieren entrada en píxeles. Segundo, el script filtra y reorganiza los archivos, seleccionando sólo aquellos que contienen múltiples objetos detectados y copiándolos en una nueva estructura de directorio organizada. Este enfoque mejora la accesibilidad y la utilidad de los datos para entrenamientos más eficientes y análisis detallados, asegurando que los datos estén listos para ser utilizados en aplicaciones avanzadas de procesamiento de imágenes.

PROCESAR LOS ARCHIVOS DE TEXTO PARA ORGANIZARLO POR ZOOM

El código propuesto tiene como objetivo principal organizar de manera eficiente archivos de datos etiquetados por niveles de zoom en un entorno de visión por computadora. Este proceso se inicia asegurando la existencia de los directorios necesarios, lo cual previene errores de acceso durante la manipulación de archivos. A continuación, el script navega a través de subdirectorios designados como test, train y valid en el directorio origen, agrupando datos específicos de anchura encontrados en archivos que indican distintos niveles de zoom.

Cada archivo es procesado para extraer y acumular las anchuras de objetos detectados, organizando estos datos por niveles de zoom en un diccionario. Posteriormente, el script crea archivos dentro de subdirectorios de destino correspondientes a cada nivel de zoom, limitando el contenido a las primeras 100 anchuras encontradas para mantener la consistencia y manejo eficiente del volumen de datos. Este enfoque garantiza que los datos estén bien organizados y sean fácilmente accesibles para análisis o entrenamiento de modelos de aprendizaje automático que dependan de la precisión de las medidas de objetos a diferentes niveles de acercamiento.

ESCRITURA DE LOS .TXT COMPLETOS PARA EL CASO DEL BUITRE NEGRO (GENERALIZABLE)

El código propuesto tiene como objetivo principal calcular y actualizar las distancias mínimas y máximas para objetos detectados en imágenes, en función de su tamaño real y el nivel de zoom aplicado. Este proceso se lleva a cabo mediante la implementación de una función que maneja archivos de datos específicamente nombrados según el nivel de zoom y ubicados en diversos subdirectorios.

La función calcular distancias realiza varios pasos críticos: inicializa una instancia de Calibrator con coordenadas geográficas específicas para ajustar la precisión del cálculo de distancias. A continuación, convierte las longitudes reales de los objetos de centímetros a metros para su uso en los cálculos. Dentro de cada subdirectorio que comienza con "zoom_", identifica y procesa los archivos de texto correspondientes, extrayendo el nivel de zoom y utilizando esta información para ajustar la longitud focal utilizada en los cálculos de distancia.

Para cada línea del archivo, calcula las distancias mínima y máxima basadas en el número de píxeles detectados y las longitudes reales proporcionadas, etiquetando cada medida con un ID único compuesto por el nivel de zoom y el número de línea. Este ID y las medidas calculadas se escriben de nuevo en el archivo, actualizando los datos originales.

Es importante destacar que este código puede adaptarse fácilmente a diferentes especies modificando los parámetros de longitud real pasados a la función y ajustando el nombre de la especie en la escritura del archivo. Esto lo hace ampliamente versátil y aplicable a diversas situaciones donde se necesite evaluar la distancia a objetos en imágenes basadas en diferentes escalas de zoom y tamaños reales.

GRAFICOS DE TAMAÑO VS DISTANCIAS

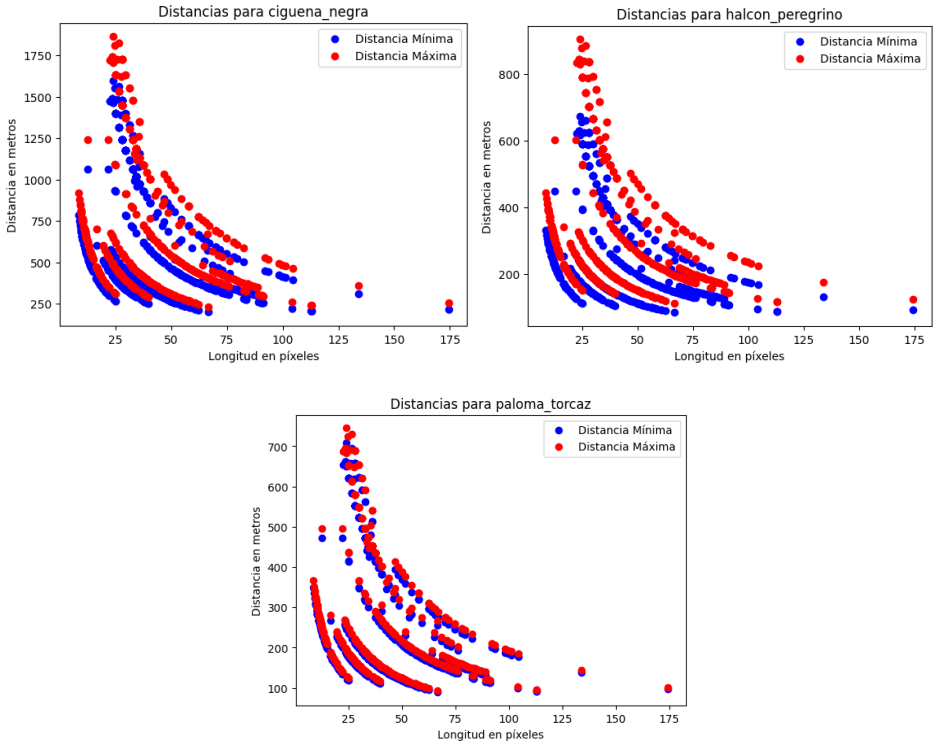


Fig.16 Gráficos comparativo cigüeña, halcón y paloma

INTERPRETACIÓN DE LA GRÁFICA DEL BUITRE NEGRO (GENERALIZABLE)

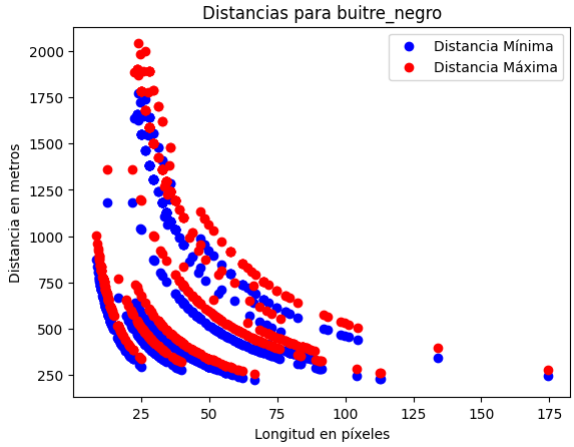


Fig.17 Gráfico Buitre

La gráfica muestra la relación entre la longitud en píxeles de aves detectadas y las distancias estimadas mínimas y máximas para la especie "buitre_negro". En el eje X, que representa la longitud en píxeles, observamos valores que van desde aproximadamente 25 hasta 175 píxeles.

En el eje Y, que muestra la distancia en metros, los valores oscilan entre aproximadamente 250 metros y poco más de 2000 metros.

Podemos ver dos series de puntos: los puntos azules representan la distancia mínima y los rojos la distancia máxima estimada para cada longitud de píxeles. Ambas series exhiben un patrón descendente claro, donde la distancia estimada, tanto mínima como máxima, disminuye a medida que aumenta la longitud en píxeles de las aves. Este patrón sugiere que a mayor tamaño aparente del ave en la imagen (mayor cantidad de píxeles), menor es la distancia estimada del ave, lo cual es coherente con la percepción visual básica.

Los puntos también muestran una forma de curva para cada serie, lo que indica una relación no lineal entre la longitud en píxeles y la distancia. Las distancias máximas tienden a ser consistentemente más altas que las mínimas para la misma longitud en píxeles, lo que es esperable dado que representan el límite superior del rango de distancia estimada para cada tamaño detectado de ave.

Esta visualización ayuda a comprender cómo varía la percepción de la distancia basada en el tamaño visual de un buitre_negro en imágenes, proporcionando información valiosa para aplicaciones de seguimiento de vida silvestre y estudios de comportamiento animal mediante técnicas de visión por computadora.

4.3 Entrenamiento

4.3.1 Segmentación de imágenes. Modelo YOLOv8.

Para entrenar el modelo YOLOv8, es necesario dar como input imágenes con sus etiquetas (aprendizaje supervisado). Es necesario que en la etiqueta de cada imagen se incluya cada uno de los objetos en la imagen (coordenadas) y la clase. Como en este proyecto se ha ideado un algoritmo de identificación y clasificación de aves en 2 pasos, la clase de las imágenes será solo 1 (ave). Posteriormente hay otros modelos que se encargan de clasificar las especies de aves, el modelo funciona para el ruido concreto con el que ha sido entrenado pero se podría aplicar el mismo procedimiento para otros lugares.
Para realizar el etiquetado de las imágenes capturadas con la cámara del proyecto, se ha hecho uso de la herramienta online Make Sense [24]. Dicha herramienta permite hacer la segmentación de un número indefinido de aves, obteniendo la etiqueta correspondiente a cada imagen. En la Figura 16 se aprecia una imagen segmentada con dicha herramienta.

Fig.18 Ejemplo de una imagen segmentada con Make Sense [24].

Antes de hacer la segmentación manual de las imágenes, fue necesario escalarlas a una resolución de 1920 x 1088 píxeles, ya que es necesario que la resolución sea divisible por 32, debido al stride del modelo. La salida que ofrece la herramienta es un fichero con extensión .txt por cada imagen que se ha segmentado. En cada uno de esos ficheros, cada línea hace referencia a cada uno de los objetos pertenecientes a la imagen. Cada línea tiene 5 números. El primero de ellos hace referencia a la clase, y los 4 siguientes sirven para identificar el cuadro delimitador de ese objeto. Los 2 primeros son la (x,y) del punto inferior izquierdo del cuadro delimitador, y los 2 siguientes son la (x,y) del punto superior derecho del cuadro delimitador. Con esos dos puntos, ya se puede obtener el cuadro delimitador completo.

$$0 - 0.021985 - 0.0279523 - 0.028894 - 0.033920$$

La expresión anterior es un ejemplo de un cuadro delimitador. Cabe destacar que las (x,y) del cuadro delimitador están contenidas en el intervalo [0,1]; esto se debe a que han sido escaladas en dicho intervalo.

Con las imágenes etiquetadas, se hicieron 3 conjuntos disjuntos:

- Conjunto de entrenamiento (80% de las imágenes totales). Conjunto usado para entrenar el modelo.
- Conjunto de validación (10% de las imágenes totales). Conjunto usado para parar el entrenamiento en caso de que haya un sobre entrenamiento y el modelo aprenda ruido de los datos de entrenamiento.
- Conjunto de test (10% de las imágenes totales). Conjunto usado para probar el modelo entrenado y realizar métricas sobre este conjunto.

Es importante remarcar que los 3 conjuntos anteriores son disjuntos, es decir que una imagen está únicamente en uno de los conjuntos. Esto se hace para evitar bias y métricas sesgadas, así como evitar sobre entrenamiento que tenga como consecuencia que el modelo no generalice.

4.3.2 Reentrenamiento mediante YOLOv11

Tras la liberación de la nueva versión de YOLO, se ha realizado un reentrenamiento con el fin de utilizar toda la potencia ofrecida por esta herramienta. A continuación, se explicará en detalle qué pasos se han seguido para ello.

Se trabaja con vídeos de 1 hora, extrayendo de dichos vídeos unos 300 frames por medio de la selección de los clips más relevantes, evitando procesar una gran cantidad de fotogramas innecesarios que no aportan información útil para el reentrenamiento del modelo. Después de seleccionar los vídeos de 1 hora , se seleccionan los clips más representativos

5 Redes Neuronales Bayesianas Explicables para la predicción de los efectos acumulativos en las aves y su hábitat

5.1 ¿Qué son los efectos acumulativos?

Los efectos acumulativos describen el impacto combinado de diversas acciones humanas y eventos naturales sobre el medio ambiente, observado generalmente a lo largo de un periodo extendido. En el contexto de la conservación de aves y sus hábitats, este concepto es crucial

para comprender cómo diferentes factores de estrés interactúan en un ecosistema y cómo pueden intensificarse mutuamente.

Consideremos, por ejemplo, el impacto del cambio climático en las aves locales. Este fenómeno puede alterar las condiciones meteorológicas de manera significativa, afectando así los ciclos de temperatura y precipitación que son vitales para la supervivencia de las especies de aves. Los cambios en la temperatura pueden desencadenar un desajuste fenológico, donde las aves migran a sus áreas de cría antes o después del momento óptimo, encontrando recursos alimenticios inadecuados para la cría de sus polluelos. Además, un aumento en la frecuencia e intensidad de eventos climáticos extremos, como tormentas y olas de calor, puede resultar en la pérdida directa de individuos y la degradación de hábitats críticos.

Estos impactos del cambio climático no ocurren de manera aislada, sino que se suman a cambios preexistentes en el entorno, como la modificación de hábitats por construcciones humanas o alteraciones en el paisaje natural. La interacción de estos efectos puede acelerar el declive de poblaciones de aves y llevar a cambios drásticos en la estructura y función del ecosistema.

Entender estos efectos acumulativos es fundamental para científicos y conservacionistas que buscan desarrollar modelos predictivos y estrategias de mitigación. Las redes neuronales bayesianas, en particular, ofrecen herramientas poderosas para integrar y analizar grandes volúmenes de datos sobre estas interacciones complejas. Estos modelos permiten generar predicciones sobre los impactos futuros del cambio climático y ayudan a formular políticas de conservación más efectivas.

5.2 ¿Qué son las redes neuronales bayesianas?

Las redes neuronales bayesianas representan una fusión de redes neuronales y estadísticas bayesianas, ofreciendo un enfoque robusto para modelar incertidumbres en sistemas de aprendizaje automático. En una red neuronal tradicional, los pesos y los parámetros suelen ser fijos una vez entrenados, mientras que, en una red neuronal bayesiana, estos parámetros se tratan como distribuciones probabilísticas, permitiendo que el modelo exprese incertidumbre sobre sus predicciones.

Este enfoque probabilístico permite que las redes neuronales bayesianas gestionen de manera más efectiva el sobreajuste y proporcionen estimaciones de confianza junto con sus predicciones, lo que es crucial en aplicaciones donde la fiabilidad de la predicción es tan importante como la predicción misma. Por ejemplo, en la ecología y la conservación, estas redes pueden prever el impacto de diferentes escenarios ambientales con una indicación clara de cuán "seguras" o "inciertas" son estas predicciones.

La integración de principios bayesianos con arquitecturas neuronales no solo mejora la interpretación de los modelos, sino que también permite una actualización continua del modelo a medida que se dispone de nuevos datos, ajustando sus predicciones para reflejar mejor la realidad observada. Esto hace que las redes neuronales bayesianas sean especialmente valiosas para tareas complejas y dinámicas donde la acumulación de datos puede cambiar la comprensión subyacente del problema en cuestión.

5.3 Modelo predictivo de la interacción Ave-Hábitat

El modelo predictivo de la interacción ave-hábitat tiene como fin principal examinar los efectos prolongados de los parques eólicos sobre las poblaciones de aves y la integridad de sus ecosistemas naturales. Este modelo se centra en evaluar cómo las infraestructuras como las turbinas eólicas pueden alterar las rutas migratorias por la creación de barreras físicas, inducir cambios en los patrones de comportamiento al evadir áreas de alto ruido o movimiento, y afectar negativamente la cadena alimenticia y los hábitats. A través de entradas detalladas como datos sobre patrones migratorios y comportamientos reproductivos de las aves, características topográficas del entorno, y ubicación operativa de los parques eólicos, el modelo intenta prever impactos como las colisiones, el desplazamiento de hábitats, y la disminución de la biodiversidad.

No obstante, la viabilidad de implementación de este modelo presenta desafíos considerables. La adquisición y el mantenimiento de extensos conjuntos de datos geoespaciales y temporales precisos son fundamentales para su funcionamiento, pero pueden ser difíciles de obtener, especialmente en zonas con recursos limitados o sin infraestructura de monitoreo ambiental establecida. Además, la modelación de probabilidades de impacto requiere una precisión en la entrada de datos que muchas veces excede las capacidades de observación y registro actuales. Estas limitaciones de datos pueden resultar en predicciones poco fiables o irrelevantes para la toma de decisiones en conservación. La complejidad de integrar y analizar variables ambientales dinámicas y multifacéticas añade otra capa de dificultad, reduciendo la aplicabilidad práctica del modelo en entornos reales donde la adaptabilidad y la precisión son cruciales para la gestión efectiva de los recursos naturales y la protección de las aves.

5.4 Monitorización y alerta en tiempo real

El modelo predictivo de monitorización y alerta en tiempo real está diseñado para evaluar los impactos de los parques eólicos sobre las aves en sus hábitats naturales, centrándose en los efectos acumulativos de largo plazo como cambios en rutas de vuelo, disponibilidad de recursos y alteración de hábitos. Utiliza entradas detalladas como datos de sensores acústicos y cámaras de video para monitorizar comportamientos y movimientos de las aves, así como micrófonos ambientales para analizar el ruido de fondo y datos meteorológicos en tiempo real que influyen en los patrones de vuelo. Además, integra información operativa de las turbinas eólicas para evaluar su interacción con el medio ambiente. El modelo genera salidas como probabilidades de colisión, desplazamiento y alteración de rutas de vuelo, además de evaluar el impacto general sobre la biodiversidad.

Sin embargo, la implementación de este modelo presenta desafíos significativos que cuestionan su viabilidad. Primero, la recopilación y análisis de datos en tiempo real exige una infraestructura tecnológica avanzada y costosa que puede ser difícil de mantener, especialmente en áreas remotas donde se localizan muchos parques eólicos. La precisión y la eficacia del modelo dependen críticamente de la calidad y la continuidad de los datos recogidos, lo cual puede ser comprometido por factores como malfuncionamientos de los sensores, limitaciones en la cobertura de las cámaras y micrófonos, y la variabilidad en las condiciones meteorológicas. Además, la integración de múltiples fuentes de datos y su análisis en tiempo real requieren capacidades computacionales y algoritmos sofisticados que pueden ser susceptibles a errores e ineficiencias. Estos factores juntos pueden limitar la aplicabilidad del modelo en entornos reales,

donde los resultados precisos son fundamentales para tomar decisiones efectivas en la conservación de las aves y la gestión de los parques eólicos. Estas barreras tecnológicas y operativas podrían resultar en predicciones inexactas o irrelevantes, afectando negativamente las estrategias de mitigación y conservación basadas en los resultados del modelo.

5.5 Barrido del cielo

Uno de los objetivos principales del proyecto IA4 BIRDS es ser capaces de mapear el cielo en su totalidad o de manera al menos parcial. Mediante los mecanismos empleados ahora mismo, somos capaces de abarcar un campo de visión horizontal de 1.98º y un campo de visión vertical de 1.14º. No obstante, el rango abarcado es relativamente pequeño en comparación con el rango total abarcable.

Cabe mencionar que el azimuth de la cámara es relativo y no absoluto. Lo que quiere decir que el norte (0º absolutos) tiene correspondencia con 144.43º relativos. Esta es una conversión que se debe de tener en consideración a la hora de realizar los cálculos y conversiones que se desarrollarán en el presente documento.

Los términos PAN, TILT, VFOV y HFOV son fundamentales para entender el funcionamiento de las cámaras en sistemas de visión, como los utilizados en el proyecto IA4Birds. PAN se refiere al movimiento horizontal de la cámara, permitiendo que gire de izquierda a derecha, mientras que TILT describe el movimiento vertical de la cámara, permitiendo que suba y baje. Estos movimientos permiten ajustar la orientación de la cámara para abarcar áreas específicas. Por otro lado, VFOV (Vertical Field of View) y HFOV (Horizontal Field of View) hacen referencia a los ángulos de visión de la cámara. El VFOV cubre el área en la dirección vertical, mientras que el HFOV lo hace en la horizontal, determinando el campo de visión que la cámara puede captar en ambas direcciones. Estos parámetros son esenciales para orientar la cámara y ajustar su cobertura en función de las necesidades del proyecto, como la monitorización de aves en IA4Birds. Unidades utilizadas

- Azimut ángulo medido en el plano horizontal contado desde el norte geográfico en el sentido de las agujas del reloj hasta la dirección de un objeto, variando su valor entre 0° y 360°.

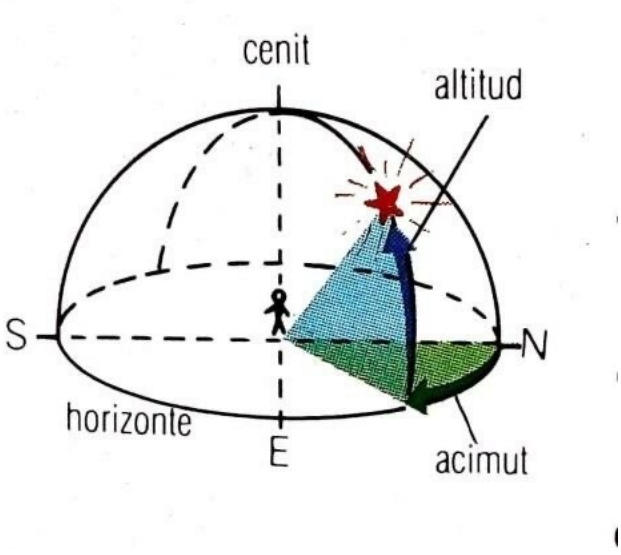


Fig.19 Ejemplo de Azimut

- Colatitud: ángulo complementario de la latitud geográfica respecto a 90°, que determina la inclinación en la esfera terrestre. Su rango varía entre 0 y 180°.

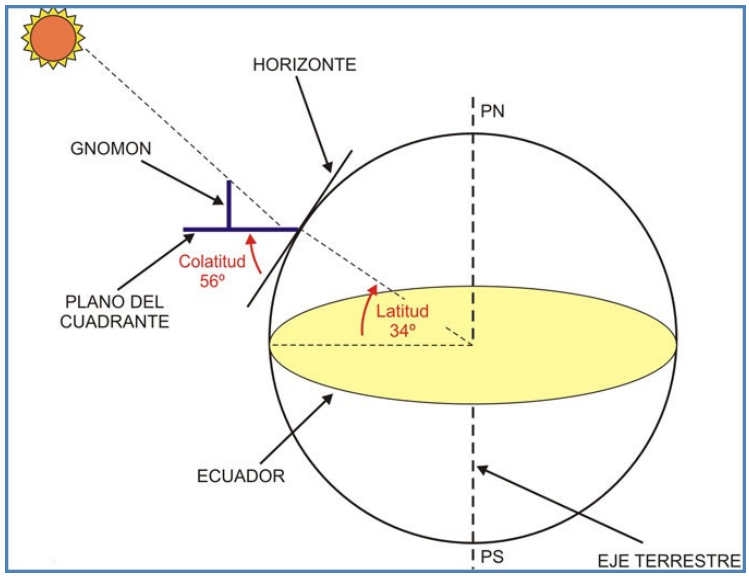


Fig.20 Ejemplo de Colatitud

- Latitud: coordenada geográfica que mide la distancia angular de un punto respecto al ecuador terrestre. Se expresa en grados con un rango de entre 0° en el ecuador y ±90° en los polos.
- Longitud: coordenada geográfica que mide la posición de un punto en la Tierra en dirección este-oeste, con respecto al meridiano de referencia de Greenwich (0° de longitud). Se expresa en grados y varía entre ±180° (este y oeste).
- Focal_Length (longitud focal): distancia entre el centro óptico de una lente y el punto focal, donde convergen los rayos de luz paralelos al eje óptico. Se mide en milímetros (mm) y determina el nivel de zoom o campo de visión (FOV) de un sistema óptico, como cámaras o telescopios.

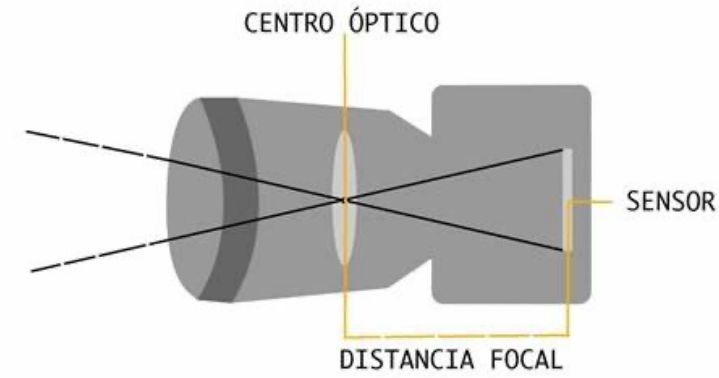


Fig.21 Ejemplo de Distancia focal

- HFOV (Campo de Visión Horizontal): ángulo que abarca la imagen en el eje horizontal cuando se captura desde una cámara o sensor óptico. Rango de [0°, 180°] en cámaras normales.

- VFOV (Campo de Visión Vertical): ángulo que abarca la imagen en el eje vertical cuando se captura desde una cámara o sensor óptico. Rango de [0°, 180°] en cámaras normales.

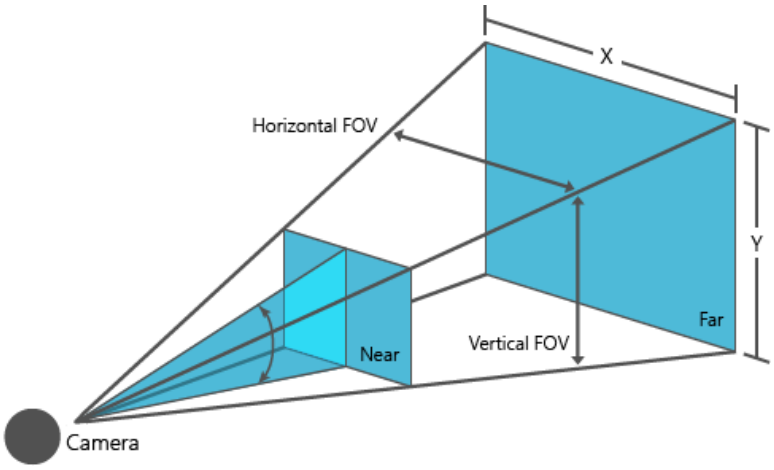


Fig.22 Ejemplo de VFOV y HFOV

5.5.1 Descripción del proceso de barrido horizontal

Se va a realizar un barrido desde los 0º absolutos hasta los 30º absolutos, es decir desde los 144.43º relativos hasta los 174.43º relativos. Se va a realizar un incremento del PAN de la cámara (HFOV) de 1.98º en 1.98º excepto en casos específicos que requieran ajustes. Cuando θ = 90º (cerca del ecuador), entonces el sin(θ) = 1 y entonces $\Delta\varphi$ = HFOV. Es decir, no hace falta corregir demasiado el paso: podemos usar incrementos fijos (por ejemplo 1.98º cada vez) y estamos cubriendo aproximadamente un mismo "arco"sobre la superficie. Sin embargo, cuando θ está cerca de 0º (polo norte) o 180º (polo sur) tiende a 0, por lo que $\Delta\varphi$ tiende a ser muy grande para cubrir el mismo arco. Dicho de otra manera, en zonas polares el paralelo se encoge y, para no "solapar" excesivamente o no perder zonas, se debe ajustar el paso usando la fórmula que se presenta a continuación. Un incremento fijo de 1.98º en esas latitudes podría cubrir un arco mayor de lo deseado:

$$\Delta\varphi(\theta) = \begin{cases} \dfrac{HFOV}{sin(\theta)} & si\ sin(\theta) \leq \sin(45) \\ 1.98 & si\ sin(\theta) > \sin(45) \end{cases}$$

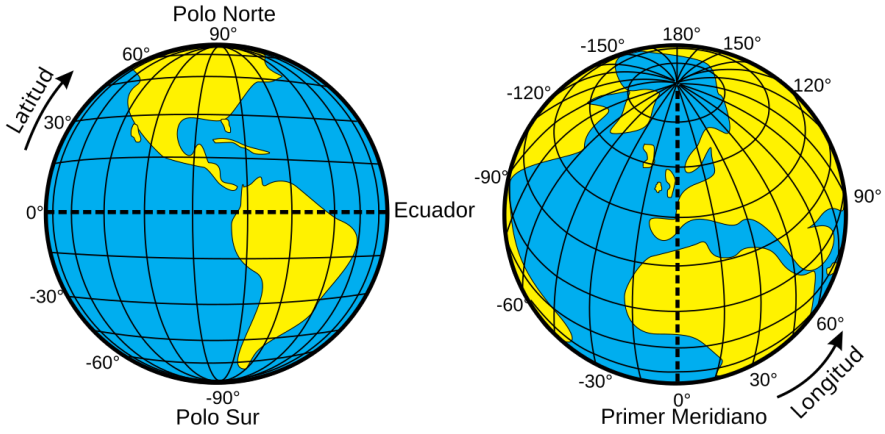


Fig.23 Latitud y longitud

Siendo $\theta = 90 - latitud$ y representando el concepto de colatitud. Además, $\Delta\varphi$ representa el incremento del azimuth relativo que es el incremento que se realizará en el PAN de la cámara medición a medición. Se puede interpretar la primera parte de la función definida a trozos como una situación la cual se da cerca de los polos, donde la colatitud es muy baja (θ es muy pequeña) o muy alta (θ se acerca a 180º). En estos casos usaremos la fórmula especificada. Se puede interpretar la segunda parte de la función definida a trozos como una situación la cual se da cerca del ecuador, donde la colatitud es aproximadamente 90º. Aquí, simplemente incrementamos el ángulo de barrido en un valor fijo de 1.98º. La razón es que el radio de los paralelos es suficientemente grande para que un incremento fijo cubra adecuadamente la región deseada sin necesidad de ajustes. Cabe destacar que los ángulos introducidos en la fórmula deben estar en radianes en vez de en grados para garantizar su correcto funcionamiento. A través del siguiente código python, se presenta la forma de aplicar la fórmula a través de la conversión previa a radianes desde grados:

```
// Ángulo en grados

theta_degrees <- 30

hfov_degrees <- 1.98

// Convertir grados a radianes

theta_radians = CONVERTIR(theta_degrees, RADIANES)

hfov_radians  = CONVERTIR(hfov_degrees, RADIANES)

// Calcular Δφ usando seno en radianes

delta_phi <- hfov_radians / SENO(theta_radians)

// Si se necesita el resultado en grados, convierte de radianes a grados

delta_phi_degrees = CONVERTIR(delta_phi, GRADOS)

IMPRIMIR("Delta φ en radianes:", delta_phi)

IMPRIMIR("Delta φ en grados:", delta_phi_degrees)
```

5.5.2 Descripción del proceso de barrido vertical

Mediante pruebas experimentales con la cámara se ha establecido un TILT inicial de 0.44 grados, lo cual se interpretará como la latitud inicial de la cámara. A continuación, se realizará un primer barrido horizontal desde 144.43º relativos hasta 174.43º relativos (0º a 30º absolutos), utilizando un incremento de PAN de 1.98º, ajustando o no según la fórmula Δϕ dependiendo de la colatitud. De cara a la realización del siguiente barrido vertical, se va a incrementar el TILT, al completar el primer barrido horizontal, se debe incrementar el TILT por el valor del VFOV. En este caso, si el VFOV tiene un valor de 1.14, entonces el nuevo TILT será $0.44 + 1.14 = 1.58$º. Con el nuevo TILT, repetir el barrido horizontal utilizando el mismo rango de azimuth relativo y el mismo incremento de PAN. A continuación, se continúa con el proceso de barrido vertical. Es decir, se continúa incrementando el TILT en 1.14º tras cada completo barrido horizontal hasta que se hayan realizado barridos verticales que cubran al menos 10 VFOV. Esto significaría realizar el proceso hasta un TILT final de aproximadamente 11.84º.

5.5.3 Duración de las grabaciones

Las grabaciones no durarán más de 1 hora ya que para el estudio del zoom y las distancias se han utilizado vídeos de 4 horas y debido a la gran cantidad de vídeos que se requieren para completar el sub-mapeo del cielo, sería inviable tomar vídeos tan largos con los que barrer la zona del cielo planteada. El número de grabaciones por barrido horizontal se calcula mediante la siguiente expresión:

$$Número\ de\ grabaciones\ horizontal\ = \frac{Rango\ total\ en\ grados}{HFOV} = \frac{30}{1.98} \approx 16$$

Respecto al número de grabaciones por barrido vertical, se realizarán 10 TILTS diferentes, el número de grabaciones verticales será de 10. El número total de grabaciones viene dado por el producto de los barridos horizontales por los barridos verticales. Viene dado por la siguiente expresión.

$$\begin{aligned} Total\ de\ grabaciones \\ = Número\ de\ grabaciones\ horizontal \\ \cdot Número\ de\ grabaciones\ vertical\ = \\ = 16 \cdot 10 = 160 \end{aligned}$$

Como se puede observar, el número de grabaciones para abarcar 30º absolutos y 10 VFOV diferentes es de 160 y si se incrementa la cantidad de horas por vídeos, se tendría una cantidad exorbitante de horas a grabar.

5.5.4 Movimiento de la cámara

```
DESDE PTZ_CONTROL IMPORTAR BasicPTZControl

controlador <- NUEVO BasicPTZControl

// Mover la cámara a un determinado PAN y TILT

pan   <- -25.0 // Ejemplo de ángulo PAN

tilt  <- -5.0  // Ejemplo de ángulo TILT

zoom  <- 5328  // Opcional: Especificar un zoom

speed <- None  // Opcional: Especificar la velocidad de movimiento

// Usar la función absolute_move para posicionar la cámara

controlador.absolute_move(pan=pan, tilt=tilt, zoom=zoom, speed=speed)
```

5.5.5 Caso de uso

Aplicando los conceptos recién discutidos. Partimos de una colatitud de 89.56º; es decir θ = 90 - 0.44º con un HFOV de 1.98º y un azimuth relativo de 155.61º. El primer paso consiste en convertir los grados de la colatitud en radianes para ser capaces de evaluar la función definida a trozos. Para ello se realiza el siguiente cálculo:

$$\theta(rad) = 89.56º \cdot \frac{\pi}{180} \approx 1.563\ radianes$$

$$45º(rad) = 45º \cdot \frac{\pi}{180} \approx 0.707\ radianes$$

Debido a que 1.563 radianes es mayor que 0.707 radianes se utiliza la segunda parte de la función definida a trozos. Para ángulos cercanos al ecuador (colatitud cercana a 90º), la fórmula que se debe utilizar es la siguiente:

$$\Delta\varphi = HFOV = 1.98º$$

Se va a ir incrementando el azimuth de 1.98º en 1.98º para cubrir todo el rango posible desde 0º hasta 30º debido a que se mantiene constante la colatitud hasta el próximo barrido vertical. Para el próximo barrido vertical se incrementa en 1.14º la colatitud y se vuelve a evaluar la función definida a trozos a ver cuál es el nuevo incremento de azimuth asociado.

5.5.6 Mapas de calor resultantes

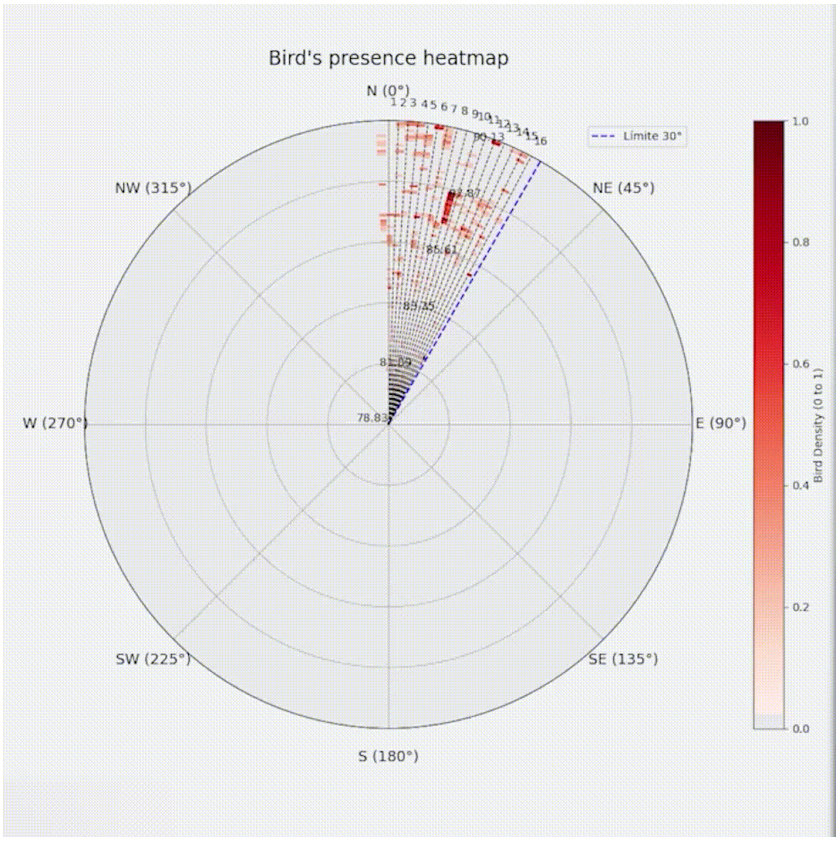


Fig.24 mapas de calor barrido del cuelo

5.5.7 Collage de densidad

En este epígrafe no se define ninguna función sino que el código aparece directamente sin encapsular. En primer lugar se define la ruta donde se ubican las imágenes overlay y la ruta donde se creará el collage general. De manera similar a lo que se ha venido haciendo en secciones anteriores se obtienen las rutas de las imágenes. Se definen los patrones Regex mediante los cuales extraer el azimut y la colatitud y se inicializa el diccionario a través del cual agrupar las imágenes por colatitud. A continuación, se iteran todas y cada una de las rutas de las imágenes, extrayendo el nombre base del archivo, el azimut y la colatitud a través de la expresión Regex definida anteriormente. Si el Regex ha tenido éxito en ambas extracciones, entonces se carga el diccionario grupos_col con la colatitud como clave del diccionario y la dupla azimut junto con la ruta de la imagen como valor. En caso de que el Regex no tenga éxito, se muestra un mensaje informativo por pantalla. A continuación, se ordenan los grupos por colatitud de forma ascendente. Se inicializan las listas vacías “filas” y “row_widths” las cuales sirven para almacenar cada fila del collage. Se iteran todos y cada uno de los elementos del diccionario cuya clave es la colatitud que acabamos de definir, por cada iteración se reordenan las imágenes por azimuth de forma ascendente y de nuevo se vuelve a recorrer el conjunto de duplas ordenadas resultante. Se carga la imagen mediante OpenCV y en caso de éxito se agrega a la lista imgs, en caso contrario se muestra un mensaje de error por pantalla. Antes de finalizar el bucle se apilan las imágenes de forma horizontal dando lugar a una fila, se agrega la fila a la lista filas y se agrega el número de columnas pertenecientes a dicha fila a la lista row_widths.
Una vez cargado row_widths se obtiene su valor máximo, se inicializa la lista filas_padded para recorrer todas y cada una de las filas. Se obtiene la altura y anchura de la fila para que en caso de que la anchura no supere el valor máximo de row_widths, entonces insertar padding a la fila y agregarla con padded a la lista filas_padded. Esta es una lista en la que todas las filas tienen la misma anchura. Por último, en caso de estar cargada la lista filas_padded se crea el collage superponiendo verticalmente las filas. Se guarda el collage en la ruta especificada y se imprime por pantalla un mensaje de éxito o error en caso contrario.

Fig.25 mapas de collage

5.6 Simulación de cambios climáticos y su impacto

5.6.1 Efectos acumulativos

El modelo de simulación de cambios climáticos y su impacto se especializa en analizar los efectos acumulativos de los cambios climáticos globales sobre los hábitats de las aves, enfocándose en las consecuencias duraderas que estos cambios tienen sobre los patrones migratorios y reproductivos. Este análisis meticuloso aborda cómo las variaciones en los patrones climáticos y estacionales, como fluctuaciones en las temperaturas, episodios de sequías y precipitaciones extremas, influyen de manera conjunta y progresiva en las rutas migratorias y los

comportamientos reproductivos, así como en la disponibilidad de recursos críticos como el alimento y las áreas de anidación. Este enfoque permite entender no solo los impactos individuales de cada variable climática, sino también cómo estos se interrelacionan y acumulan con el tiempo, afectando profundamente la ecología y el comportamiento de las aves en sus entornos naturales.

5.6.1 Funcionalidad de la red neuronal bayesiana

Entradas del modelo

El modelo de red neuronal bayesiana diseñado para la predicción de los efectos del cambio climático sobre la biodiversidad avícola se basa en una serie de entradas detalladas que permiten simular las interacciones entre las condiciones climáticas y las respuestas de las aves. Estas entradas incluyen:

Registros de temperatura y proyecciones futuras: Esta entrada incluye el monitoreo de las tendencias en el aumento o disminución de las temperaturas a lo largo del tiempo, permitiendo la predicción de condiciones climáticas futuras y sus impactos sobre los ecosistemas avícolas.

Información sobre sequías y precipitaciones extremas: El análisis de la frecuencia e intensidad de sequías y lluvias torrenciales proporciona información sobre cómo estos fenómenos climáticos afectan la disponibilidad de recursos naturales, como fuentes de agua y alimentos, así como las condiciones de hábitat para las aves.

Olas de calor: Se documenta la frecuencia y severidad de olas de calor, que influyen directamente en el bienestar de la fauna y flora local. Este dato es crucial, ya que los episodios extremos de calor pueden alterar el comportamiento y las condiciones de vida de las aves, afectando su supervivencia y distribución.

Salidas del modelo

Basado en las entradas proporcionadas, el modelo genera varias salidas clave que permiten hacer predicciones sobre los efectos del cambio climático en las poblaciones de aves. Estas salidas incluyen:Probabilidad de desplazamiento de hábitat: Esta salida calcula la probabilidad de que las aves alteren su ubicación habitual en respuesta a los cambios en las condiciones climáticas. El modelo estima la posible migración de especies hacia nuevas áreas para sobrevivir a las alteraciones del hábitat causadas por fenómenos como el aumento de la temperatura o la escasez de recursos.
Impacto en la biodiversidad avícola: Esta predicción estima la reducción en la diversidad de aves, reflejando las probabilidades de alteraciones en los ecosistemas avícolas debido al cambio climático. Los cambios en las condiciones climáticas podrían llevar a la extinción local de especies o al desplazamiento hacia otras regiones.
Cambios en patrones migratorios y reproductivos: El modelo genera probabilidades de alteraciones significativas en las rutas migratorias y los comportamientos reproductivos de las aves, debido al cambio climático. Esto incluye cambios en los tiempos de migración y los períodos de anidación, que podrían verse desajustados con los ciclos naturales de las aves.

Viabilidad

La viabilidad de implementar este modelo está reforzada por el uso de tecnologías avanzadas como las redes neuronales bayesianas y el aprendizaje no supervisado, que ofrecen un análisis sofisticado y profundo de grandes volúmenes de datos climáticos y biológicos. Estas tecnologías proporcionan la flexibilidad y capacidad necesaria para adaptarse a la compleja dinámica de los ecosistemas, permitiendo una implementación efectiva en diversos entornos y condiciones. La habilidad de estas herramientas para manejar y analizar datos de manera eficiente garantiza que el modelo puede ser desplegado efectivamente, ofreciendo predicciones precisas y relevantes para la conservación y manejo de la biodiversidad avícola frente al cambio climático.

Ejemplo

En el estudio realizado en Castilla y León, España, se utilizó un modelo de simulación para evaluar los impactos combinados de factores climáticos y ambientales sobre la avifauna local. Este modelo incorpora registros de temperatura que indican un aumento promedio de 0.5°C por década, sugiriendo posibles cambios en los tiempos de migración y reproducción de las aves. También se analiza la variabilidad en las precipitaciones, resaltando un aumento en la frecuencia de sequías y eventos de lluvias torrenciales en los últimos 20 años, que podrían alterar las áreas de alimentación y anidación al modificar las condiciones de las zonas tradicionalmente secas o inundarlas. Adicionalmente, se integran datos sobre olas de calor, con un registro de diez episodios extremos durante el último verano, donde las temperaturas superaron los 40°C, incrementando potencialmente el estrés térmico en las aves y afectando su supervivencia y comportamiento natural. Basándose en estos datos, el modelo predice un desplazamiento significativo de hábitats, estimando que un 25% de las aves podrían modificar sus áreas habituales en respuesta a los cambios climáticos. Se anticipa además una reducción del 15% en la diversidad de especies avícolas, reflejando la afectación de los hábitats y los efectos acumulativos del cambio climático. Se calcula también que aproximadamente un 30% de las aves ajustarán sus patrones migratorios y reproductivos debido al incremento de las temperaturas, lo que subraya la importancia de incorporar estos datos en la planificación de la conservación de la biodiversidad avícola.

Este modelo es un prototipo ideado dentro del proyecto IA4Birds y no ha sido completamente desarrollado o implementado. Aunque se ha conceptualizado y se han utilizado simulaciones basadas en datos históricos, no se ha llevado a cabo un desarrollo práctico del modelo en un entorno real de monitoreo. El propósito del modelo es proporcionar una base para futuras investigaciones y para la integración de tecnologías avanzadas en la protección de la biodiversidad avícola, pero aún se encuentra en fase de prueba y validación.

6 Referencias

[1]. M. Ramashini, P. E. Abas, K. Mohanchandra y L. C. D. Silva, ((Robust cepstral feature for bird sound classification,)) International Journal of Electrical and Computer Engineering (IJECE), vol. 12, n.o 2, págs. 1477-1487, 1 de abr. de 2022, Number: 2, issn: 2722-2578. doi: 10.11591/ijece.v12i2.pp1477- 1487. dirección: https://ijece.iaescore.com/index.php/IJECE/article/view/25893 (visitado 17-05-2024).

[2]. S. Carvalho, ((Automatic classification of bird sounds: Using MFCC and mel spectrogram features with deep learning,)) Vietnam Journal of Computer Science, dirección: https://www.academia.edu/114461733/Automatic_Classification_of_Bird_Sounds_Using_MFCC_and_Mel_Spectrogram_Features_with_Deep_Learning (visitado 17-05-2024).

[3]. Venkatesh, S.; Moffat, D.; Miranda, E.R. You Only Hear Once: A YOLO-like Algorithm for Audio Segmentation and Sound Event Detection. Appl. Sci. 2022, 12, 3293. https://doi.org/10.3390/app12073293

[4]. xeno-canto :: Sharing wildlife sounds from around the world, dirección: https://xeno-canto.org/ (visitado 24-09-2024).

[5]. API :: xeno-canto., dirección: https://xeno-canto.org/explore/api (visitado17-05-2024).

[6]. J. Alves, J. Shamoun-Baranes, P. Desmet y col., Monitoring continent-wide aerial patterns of bird movements using weather radars. 31 de mar. de 2015.

[7]. F. Liechti y H. van Gasteren, CURRENT STAGE OF BIRD RADAR SYSTEMS, 1 de jun. de 2010.

[8]. P. Gemmar, Detection of bird activity in radar images, 1 de ene. de 2012. dirección: https://www.academia.edu/71442826/Detection_of_Bird_Activity_in_Radar_Images (visitado 17-05-2024).

[9]. S. Albawi, T. A. Mohammed y S. Al-Zawi, ((Understanding of a convolutional neuralnetwork,)) en 2017 International Conference on Engineering and Technology (ICET), ago. de 2017, págs. 1-6. doi:10.1109/ICEngTechnol.2017.8308186. dirección: https://ieeexplore.ieee.org/document/8308186 (visitado17-05-2024).

[10]. S. Indolia, A. K. Goswami, S. P. Mishra y P. Asopa, ((Conceptual Understanding of Convolutional Neural Network-A Deep Learning Approach,)) Procedia Computer Science, International Conference on Computational Intelligence and Data Science, vol. 132, págs. 679-688, 1 de ene. de 2018, issn: 1877-0509. doi:10.1016/j.procs .201 .0. 069. dirección: https://www.sciencedirect.com/science/article/pii/S1877050918308019 (visitado 17-05-2024).

[11]. J. Nagi, F. Ducatelle, G. A. Di Caro y col., Max-pooling convolutional neural networks for vision-based hand gesture recognition, en 2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), nov. de 2011, págs. 342-347. doi:10.1109/ICSIPA.2011.6144164. dirección: https://ieeexplore.ieee.org/document/6144164 (visitado 17-05-2024).

[12]. S. R. Dubey, S. K. Singh y B. Chaudhuri, Activation Functions in Deep Learning: A comprehensive Survey and Benchmark, Neurocomputing, vol. 503, 1 de jul. de 2022. doi: 10.1016/j.neucom.2022.06.111.

[13]. K. He, X. Zhang, S. Ren y J. Sun, Deep Residual Learning for Image Recognition, 10 de dic. de 2015. doi: 10.48550/arXiv.1512.03385. arXiv: 1512.03385[cs]. dirección: http://arxiv.org/abs/1512.03385 (visitado 20-05-2024).

[14]. G. Huang, Z. Liu, L. van der Maaten y K. Weinberger, Densely Connected Convolutional Networks. 24 de jul. de 2017. doi: 10.1109/CVPR.2017.243.

[15]. C. Shorten y T. M. Khoshgoftaar, A survey on Image Data Augmentation for Deep Learning, Journal of Big Data, vol. 6, n.o 1, pág. 60, 6 de jul. de 2019, issn: 2196-1115. doi: 10.1186/s40537-019-0197-0. dirección: https://doi.org/10.1186/s40537-019-0197-0 (visitado 20-05-2024).

[16]. A. Yang y D. Silver, The Disadvantage of CNN versus DBN Image Classification Under Adversarial Conditions, 18 de mayo de 2021. doi: 10.21428/594757db.b65acd40.

[17]. M. Tan y Q. V. Le, EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, 11 de sep. de 2020. doi: 10.48550/arXiv.1905.11946. arXiv: 1905.11946[cs,stat]. dirección: http://arxiv.org/abs/1905.11946 (visitado 17-05-2024).

[18]. A. Vaswani, N. Shazeer, N. Parmar y col., Attention is all you need, 1 de ago. de 2023. arXiv: 1706.03762[cs]. dirección: http://arxiv.org/abs/1706.03762 (visitado 15-05-2024).

[19]. A. Gillioz, J. Casas, E. Mugellini y O. Abou Khaled, Overview of the Transformer-based Models for NLP Tasks. 26 de sep. de 2020, 179 págs., Pages: 183. doi: 10.15439/2020F20.

[20]. A. Dosovitskiy, L. Beyer, A. Kolesnikov y col., An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, 3 de jun. de 2021. doi: 10.48550/arXiv.2010.11929. arXiv: 2010.11929[cs]. dirección: http://arxiv.org/abs/2010.11929 (visitado 17-05-2024).

[21]. M. Ning, Y. Lu, W. Hou y M. Matskin, YOLOv4-object: an Efficient Model and Method for Object Discovery, en 2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC), Madrid, Spain: IEEE, jul. de 2021, págs. 31-36, isbn: 978-1-66542-463-9. doi: 10.1109/COMPSAC51774.2021.00016. dirección: https://ieeexplore.ieee.org/document/9529473 (visitado 17-05-2024).

[22]. L. Lucchese y S. Mitra, Color Image Segmentation: A State-of-the-Art Survey, Proceedings of Indian National Science Academy, vol. 2, 1 de ene. de 2001.

[23]. AXIS Q6225-LE PTZ Camera Axis Communications, dirección: https://www.axis.com/es-es/products/axis-q6225-le (visitado 19-09-2024).

[24]. Make Sense. dirección: https://www.makesense.ai/ (visitado 16-05-2024).