Frailejon Detection (a.k.a. 'Big Monk' Detection) bajo metodología CRISP

[ac] Camilo Andrés González Vargas, [ac] Juan David Valencia Sandoval, [ac] Paula Alejandra Velasco Cuberos, [ac] Pavel Mauricio Dussan Gutiérrez, [ac] Tatiana Roa Ahumada, [ac] Yuri Angélica Reina Guzmán, [bc] Sergio Alberto Mora Pardo

a. Estudiante de Maestría en Analítica para la Inteligencia de Negocios
b. Profesor, Departamento de Ingeniería Industrial
c. Pontificia Universidad Javeriana, Bogotá, Colombia

BUSINESS UNDERSTANDING

Background:

Las redes neuronales, inspiradas en el funcionamiento del cerebro humano, son modelos computacionales capaces de aprender a partir de datos y realizar tareas complejas. En el contexto de la detección de frailejones, estas redes actúan como un "cerebro artificial" que aprende a reconocer patrones visuales característicos de estas plantas en imágenes aéreas.

Las redes neuronales convolucionales (CNN), especialmente diseñadas para el procesamiento de imágenes, son la herramienta adecuada para esta tarea. Estas redes están compuestas por capas de neuronas interconectadas que procesan la información de manera jerárquica. Las primeras capas extraen características básicas de la imagen, como bordes y texturas, mientras que las capas posteriores combinan estas características para identificar objetos más complejos, como los frailejones.

El entrenamiento de una CNN implica presentarle miles de imágenes etiquetadas, donde se indica la ubicación de los frailejones. A medida que la red procesa estas imágenes, ajusta los pesos de sus conexiones para minimizar el error entre sus predicciones y las etiquetas reales. Este proceso iterativo permite a la red aprender a reconocer frailejones en nuevas imágenes, cumpliendo con el objetivo de este trabajo.

Estudios recientes han demostrado el potencial de las técnicas de teledetección, incluyendo el análisis de imágenes basado en objetos (Ruiz-Olaya et al., 2018) y el aprendizaje profundo (García-Llamas et al., 2020), para la identificación y mapeo de frailejones en los Andes colombianos. Sin embargo, la mayoría de estos trabajos se han centrado en áreas específicas o han utilizado imágenes satelitales de menor resolución.

El monitoreo y preservación del ecosistema de los páramos ha sido un foco importante de investigación debido a su relevancia ambiental. Según la CUC (2024), los páramos juegan un papel clave en la regulación del ciclo hidrológico y la captación de carbono en regiones montañosas. La aplicación de tecnologías avanzadas como el aprendizaje profundo, en combinación con imágenes áreas de alta resolución, puede facilitar la implementación de estrategias de conservación más precisas y eficaces, permitiendo identificar áreas con densidades bajas de frailejones y así dirigir acciones de restauración.

Este proyecto busca avanzar en este campo mediante la aplicación de redes neuronales convolucionales (CNN) para la detección automática de frailejones en imágenes aéreas de alta resolución de los páramos de Chingaza y Cruz Verde. Al aprovechar la capacidad de las CNN para aprender características complejas y adaptarse a la variabilidad de los datos, esperamos lograr una detección precisa y eficiente de frailejones en estos ecosistemas.

Determine Business objectives

Business goal:

BG1: Impulsar la preservación y monitoreo del ecosistema de los páramos Chingaza y Cruz Verde mediante la propuesta de una metodología para la reconstrucción del ecosistema en zonas de baja densidad de poblaciones de frailejones de la especia *Espeletia.*

Business success criteria:

BG1 – KPI1: Identificar las zonas de baja densidad poblacional de frailejones según el área de influencia (cálculo poblacional).

Determine Data mining goals

Data mining goal:

DMG1: Construir una red neuronal para la detección automática de frailejones sobre imágenes aéreas del páramo de Chingaza y Cruz Verde

Data mining success criteria:

DMG 1 – KPI1: Diseñar cinco (5) propuestas de redes con diferentes arquitecturas y/o hiperparámetros

DMG 1 – KPI2: Obtener un desempeño en los modelos medido por AUC de la siguiente forma:

- Red Sencilla: > 0.88
- Multi-capa: > 0.95
- Convolucional: > 0.95
- Convolucional + VGG16 > 0.99

DMG 1 – KP3: Verificar de forma visual (cualitativa) la detección de frailejones por medio del mejor modelo

DATA UNDERSTANDING

Se cuenta con 2 dataset 'Frailejon' con 21 imágenes y 'NoFrailejon' con 29 imágenes, lo que sugiere un ligero desbalance en las clases. Las imágenes están en formato RGB y tienen el mismo tamaño, lo que es ideal para su procesamiento en modelos de clasificación o segmentación de imágenes.

Figura 1. Visualización de ejemplos de dataset

Se realizó un análisis de distribución de intensidades para determinar si hay una mayor presencia de ciertos colores en las imágenes de frailejones en comparación con las imágenes que no los contienen.

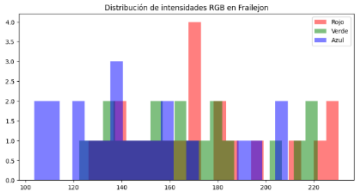

Figura 2. Distribución de intensidades RGB en Frailejón

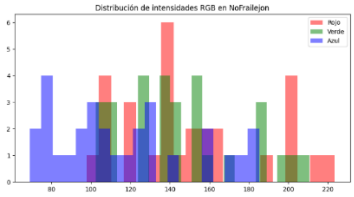

Figura 3. Distribución de intensidades RGB en 'NoFrailejon'

Se evidencia que las imágenes del dataset Frailejon tienen una mayor presencia de tonos verdes, lo que es consistente con la vegetación de los frailejones, mientras que el canal azul es menos prominente. Las imágenes de NoFrailejon muestran una mayor variabilidad en los tonos, especialmente en los canales rojo y azul, lo que podría reflejar la presencia de otros objetos o áreas del paisaje que no contienen vegetación densa.

Se aplicó la técnica de detección de bordes para identificar áreas donde ocurren cambios abruptos en la intensidad de los píxeles, lo que permite resaltar los límites de los objetos presentes.

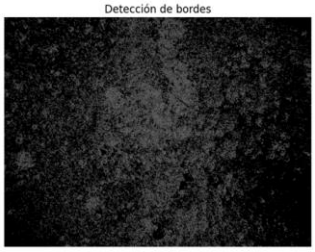

Figura 4. Detección de bordes

La imagen muestra una gran cantidad de bordes, lo que indica una alta variabilidad en la textura del terreno y la vegetación. Sin embargo, los bordes detectados no presentan contornos claros de objetos, lo que sugiere una transición suave entre las áreas de la imagen.

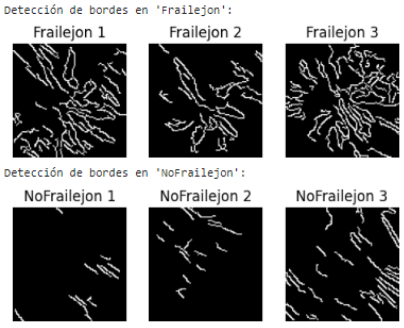

Figura 5. Detección de bordes dataset

En las imágenes de NoFrailejon, los bordes detectados son en su mayoría finos y dispersos. Lo que sugiere que las imágenes contienen estructuras sin contornos claramente definidos o que están más relacionadas con cambios graduales en la intensidad, mientras que las imágenes de Frailejon muestran bordes más densos y con formas más definidas, estas diferencias podrían ser útiles para distinguir entre los dos tipos de imágenes. Se realizó un análisis de contornos para identificar las formas de los objetos presentes en la imagen.

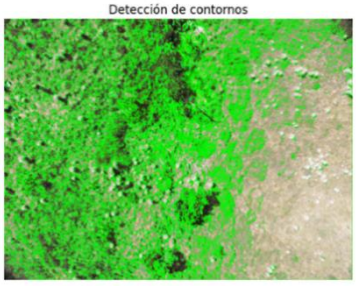

Figura 6. Detección de contornos

Los contornos resaltan principalmente las áreas con vegetación, lo que sugiere que estas zonas tienen una estructura más compleja y bien definida. En las áreas de menor vegetación, los contornos son menos prominentes.

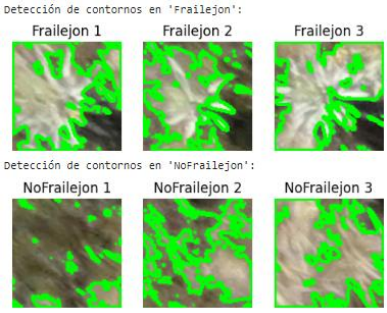

Figura 7. Detección de contornos dataset

Las imágenes de Frailejon muestran contornos más numerosos y bien definidos alrededor de la estructura de las plantas. Los frailejones, por su morfología, tienen hojas alargadas y distintivas, lo que genera bordes más notables y complejos. Mientras que, en las NoFrailejon, aunque hay una detección de contornos, no son tan complejos o definidos como los de los frailejones. Los contornos parecen seguir patrones menos estructurados, lo que podría estar relacionado con elementos del paisaje, como rocas o terreno sin vegetación.

DATA PREPARATION

Para alcanzar los objetivos propuestos, se llevaron a cabo diferentes procedimientos de preparación y preprocesamiento de las imágenes del dataset original 'data_F'. Estos procedimientos varían en función del tipo de red que se va a entrenar. A continuación, se destacan las etapas más relevantes:

División de imágenes en 5 partes: Sobre las 50 imágenes originales (21 clasificadas como 'Frailejon' y 29 como 'NoFrailejon'), se realiza una división en cinco (5) partes. De esta forma se cuenta con 250 imágenes y una población más grande de análisis. Se realiza por medio de la función 'Import_Imagenes()' desarrollada en el enunciado del proyecto.

División entre datos de entrenamiento y prueba: En todos los casos, se realizó una división del conjunto de 250 imágenes en una proporción del 70%-30%. Así, inicialmente, el conjunto de entrenamiento contiene 175 imágenes, mientras que el de prueba cuenta con 75 imágenes.

Normalización de los valores RGB: Las imágenes están en formato de color de tres canales (RGB), con valores que oscilan entre 0 y 256 para cada canal. Para normalizar estos valores, se dividen entre 256, de manera que los datos se encuentren en un rango de 0 a 1, representados como números decimales. Este proceso se aplica en todos los casos.

Reescalado de las imágenes: Para los modelos basados en transferencia de aprendizaje, es necesario ajustar el tamaño de las imágenes. Originalmente, el dataset 'data_F' contiene imágenes de 70 x 70 píxeles, sin embargo, los modelos pre entrenados como lo son el VGG16 o ResNet50 requieren de un input de imágenes de 224 x 224 pixeles. De esta forma, se genera la transformación por medio de la función 'Resizing' de la librería de tensorflow en Pyhton.

Data Augmentation: Se desarrolló un procedimiento de aumento de datos, donde, por medio de las imágenes originales, se recrean nuevos elementos al rotarlos, desplazarlos verticalmente/horizontalmente, hacerles 'zoom' y recortes. Se utilizará este nuevo dataset para la mejora de desempeño y lograr las metas definidas como criterios de éxito.

Dataset description:

Las vistas minables trabajadas para el desarrollo del modelo son las siguientes:

Modelo 1- Red neuronal sencilla: Se cuenta con un arreglo de 175 imágenes contra 14.700 características para entrenamiento y 75 imágenes contra 14.700 características para prueba. Adicionalmente, un arreglo del número de imágenes de entrenamiento y prueba junto a la clasificación de 'frailejón' o 'no frailejon' codificado con 1 y 0 respectivamente.

En la figura 8, se observa la vista minable para el primer modelo.

```
Atributos Entrenamiento: (14700, 175)
 Atributos Validación: (14700, 75)
 Clase Entrenamiento: (1, 175)
 Clase Validación: (1, 75)
```

Figura 8. Vista minable modelo 1

Modelo 2 - Red neuronal multicapa: Se cuenta con un arreglo de 32 lotes de 32 imágenes con contenido de 70 x 70 pixeles en 3 canales de color RGB para entrenamiento; el set de prueba se mantiene de 75 imágenes con formato de 70 x70 pixeles y 3 canales. Adicionalmente, un arreglo del número de imágenes de entrenamiento y prueba junto a la clasificación de 'frailejón' o 'no frailejon' codificado con 1 y 0 respectivamente.

En la figura 9, se observa la vista minable para el perceptrón multicapa:

```
Atributos entrenamiento (lotes de 32 imágenes): (32, 70, 70, 3)
Clase Entrenamiento (lotes de 32 imágenes): (32, 1)
Atributos Validación: (75, 70, 70, 3)
Clase Validación: (75, 1)
```

Figura 9. Vista minable modelo 2

Modelo 3 - Red neuronal Convolucional: Se cuenta con un arreglo de 175 imágenes con contenido de 70 x 70 pixeles en 3 canales de color RGB para entrenamiento y de 75 para prueba. Adicionalmente, un arreglo del número de imágenes de entrenamiento y prueba junto a la clasificación de 'frailejón' o 'no frailejon' codificado con 1 y 0 respectivamente.

En la figura 10, se observa la vista minable para la red neuronal convolucional:

```
Atributos Entrenamiento: (175, 70, 70, 3)
 Atributos Validación: (75, 70, 70, 3)
 Clase Entrenamiento: (175, 1)
 Clase Validación: (75, 1)
```

Figura 10. Vista minable modelo 3

Modelo 4 – Red Neuronal Convolucional + VGG16 y Modelo 5 – Red Neuronal Convolucional + ResNet50: Se cuenta con un arreglo de 32 lotes de 32 imágenes con contenido de 224 x 224 pixeles en 3 canales de color RGB para entrenamiento; el set de prueba se mantiene de 75 imágenes con formato de 224 x 224 pixeles y 3 canales. Adicionalmente, un arreglo del número de imágenes de entrenamiento y prueba junto a la clasificación de 'frailejón' o 'no frailejon' codificado con 1 y 0 respectivamente.

En la figura 11, se observa la vista minable para las redes neuronales convolucionales con transferencia de aprendizaje:

```
Atributos entrenamiento (lotes de 32 imágenes): (32, 224, 224, 3)
Clase Entrenamiento (lotes de 32 imágenes): (32, 1)
Atributos Validación: (75, 224, 224, 3)
Clase Validación: (75, 1)
```

Figura 11. Vista minable modelo 4 y 5

MODELING

A continuación, se describe brevemente cada una de las arquitecturas empleadas:

Modelo 1- Red neuronal sencilla: Para este modelo, la red neuronal cuenta con solo 2 capas, la capa de entrada y la capa de salida. La capa de entrada estará compuesta por un número a definir de neuronas y la capa final estará compuesta por una única neurona dada la naturaleza del problema de clasificación binaria.

En relación con la primera capa, sus parámetros son los siguientes: Función de activación: ReLu, Kernel para inicialización de los pesos: GlorotUniform, Inicializador de los sesgos: función Zeros de la librería Numpy, input_dim = 14700.

En relación con la capa final, su único parámetro será la función de activación: Sigmoid.

En la compilación del modelo, se define como optimizador: Adam, la función de pérdida: binary_crossentropy, y la métrica para evaluar al modelo: AUC.

En la figura 12, se visualiza la arquitectura del modelo.

Layer (type)	Output Shape	Param #
dense_12 (Dense)	(None, 56)	823,256
dense_13 (Dense)	(None, 1)	57

Figura 12. Ejemplo de arquitectura del modelo 1

Modelo 2 - Red neuronal multicapa: A diferencia del modelo 1, se añadirán varias capas ocultas entre la capa de entrada y la capa de salida.

Adicionalmente, con el propósito de evitar problemas de sobreajuste y de estabilizar y acelerar el entrenamiento, se utilizarán las técnicas de DropOut y de BatchNormalization respectivamente, justo después de la capa inicial y las capas de activación. Esto se puede evidenciar en la arquitectura del modelo referenciado en el numeral 2 del Anexo A. Para la compilación del modelo 2, se utilizan los mismos parámetros del modelo 1.

Para las capas ocultas, se realizarán diferentes ensayos con distintos números de neuronas (units). La función de activación será la función ReLu. En cuanto a las inicializaciones de los pesos: GlorotUniform y el inicializador de los sesgos: función Zeros de la librería Numpy.

Para las capas asociadas a la técnica de DropOut, se tanteará diferentes valores entre 0 y 1, y las asociadas a la técnica de BatchNormalization(), se dejarán los parámetros que tenga por defecto.

Modelo 3 - Red neuronal Convolucional: En cuanto a este modelo, se busca aprovechar la capacidad de las capas convolucionales. Estas capas dedican neuronas en concreto a analizar un grupo de pixeles a través de filtros e identificando patrones. Como se visualiza en la figura 13, las capas convolucionales se sitúan previo a una red neuronal *full-connected* culminando su procesamiento con una capa de aplanado (flatten).

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 35, 35, 32)	4,736
max_pooling2d (MaxPooling2D)	(None, 17, 17, 32)	0
conv2d_1 (Conv2D)	(None, 9, 9, 64)	100,416
max_pooling2d_1 (MaxPooling2D)	(None, 4, 4, 64)	0
flatten (Flatten)	(None, 1024)	0
dense (Dense)	(None, 1024)	1,049,600
batch_normalization (BatchNormalization)	(None, 1024)	4,096
dropout (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 512)	524,800
batch_normalization_1 (BatchNormalization)	(None, 512)	2,048
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 64)	32,832
batch_normalization_2 (BatchNormalization)	(None, 64)	256
dense_3 (Dense)	(None, 1)	65

Figura 13. Ejemplo de arquitectura del modelo 3

La configuración de los parámetros de la capa inicial y final se realiza igual que la del modelo 1. La configuración de la red neuronal full-connected, es la misma configuración que el modelo 2. Para las capas convolucionales, se tantearán un número de filtros (filters) y sus correspondientes tamaños (kernel_size), se definirán un zero padding, ReLu como la función de activación y un stride de 2. En cuanto a las inicializaciones de los pesos: 'he_normal' y el inicializador de los sesgos: función Zeros de la librería Numpy.

La compilación de este modelo usa la misma configuración que el modelo 1.

Modelo 4 – Red Neuronal Convolucional VGG16: Para este modelo se quiere sacar provecho de una arquitectura de red neuronal convolucional (CNN) desarrollada por el equipo de Visual Geometry Group (VGG) de la Universidad de Oxford, la cual se destaca por su precisión en tareas de clasificación de imágenes.

Se sustituyen las capas convolucionales desarrolladas por la arquitectura VGG16 como se visualiza en la figura 14. De esta manera se estaría aplicando un método de transferencia de aprendizaje.

Para configuración de la arquitectura VGG16, se define los parámetros weights='imagenet', include_top=False para evitar que se modifiquen los pesos y sesgos de la arquitectura, e input_shape=(224, 224, 3). Es necesario además añadir una capa de aplanado posterior a la arquitectura VGG16 dado por el parámetro include_top=False. Finalmente, se define una capa oculta seguida por una capa de salida de clasificación binaria por medio de la función de activación 'Sigmoid'.

El modelo culmina con un proceso de compilación al igual que los modelos anteriores.

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(1, 7, 7, 512)	14,714,688
flatten (Flatten)	(1, 25088)	0
dense (Dense)	(1, 1024)	25,691,136
dense_1 (Dense)	(1, 1)	1,025

Figura 14. Ejemplo de arquitectura del modelo 4

Modelo 5 – Red Neuronal Convolucional + ResNet50: Residual Network con 50 capas (ResNet50) es una red neuronal profunda desarrollada por el equipo de Microsoft Research. En este caso, se sustituyen las capas convolucionales desarrolladas por la arquitectura

ResNet50 como se visualiza en la figura 15 y se realiza la transferencia de aprendizaje.

Inicilamente, se define el modelo con los parámetros weights='imagenet', include_top=False para evitar que se modifiquen los pesos y sesgos de la arquitectura, e input_shape=(224, 224, 3). Es necesario además añadir una capa de aplanado al igual que con VGG16. Finalmente, para la sección 'Full Connected' se desarrolla una capa oculta, seguida por un 'Batch Normalization' y una regularización por 'Dropout'.

La arquitectura es la siguiente:

Layer (type)	Output Shape	Param #
resnet50 (Functional)	(1, 7, 7, 2048)	23,587,712
flatten_2 (Flatten)	(1, 100352)	0
dense_6 (Dense)	(1, 1024)	102,761,472
batch_normalization_3 (BatchNormalization)	(1, 1024)	4,096
dropout_2 (Dropout)	(1, 1024)	0
dense_7 (Dense)	(1, 1)	1,025

Figura 15. Ejemplo de arquitectura del modelo 5

El modelo culmina con un proceso de compilación al igual que los modelos anteriores.

Teniendo en cuenta que la red neuronal ResNet50 fue preentrenada con base en el conjunto de datos ImageNet, el cual contiene imágenes de gran cantidad de paisajes naturales, es una opción apropiada en la detección de frailejones pues ha aprendido a diferenciar características asociadas a texturas, bordes, formas, etc., que pueden ser reutilizadas basados en la naturaleza del proyecto. De esta forma, los patrones visuales en la tarea base, asociada a la clasificación de imágenes en ImageNet y la tarea objetivo, que comprende la detección de frailejones, se encuentran fuertemente relacionadas; así, los pesos enlazados a las características generales aprendidas en la red ResNet50, fundamentan un modelo de aprendizaje valioso para la necesidad planteada.

EVALUATION:

Se aplicaron 5 modelos, para los cuales se calculó el AUC como métrica de evaluación. Adicionalmente se calcula la función de pérdida binary_crossentropy para determinar el rendimiento en los modelos.

A continuación, se realiza la descripción de los modelos a partir de los resultados obtenidos con base en las métricas anteriormente indicadas:

Modelo 1- Red neuronal sencilla: Se evaluaron varias configuraciones utilizando diferentes hiperparámetros, con 8 épocas de entrenamiento y variaciones en el número de neuronas en la capa oculta: 8, 40, 48 y 56. La mejor configuración con 40 neuronas, alcanzó un **AUC** de 0,8851 y una **pérdida (loss)** de 0,4371 en validación.

Modelo 2 - Red neuronal multicapa: al aplicar los diferentes métodos para mejorar el rendimiento se obtuvo los siguientes resultados:

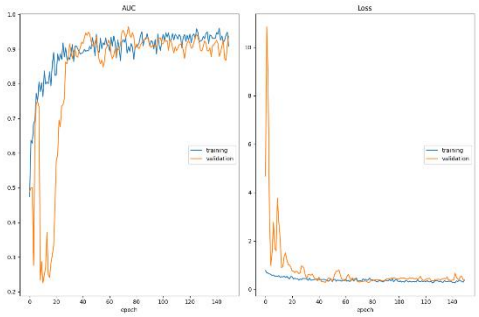

Figura 16. Resultados AUC – Modelo 2

Con 150 épocas, 4092 neuronas, se aplica una capa flatten. Se obtuvo un **AUC**: 0,960 y una **pérdida** de: 0,262 en training y en validación **AUC**: 0.964 y **pérdida**: 0.287. Se realiza la configuración de los parámetros para un Grid Search: # capas: [2, 3, 4, 5]; # neuronas: [64, 128, 256, 512] y dropout: [0.1, 0.2, 0.3, 0.4]. Se obtiene un modelo # capas: 2; # neuronas: 512 y dropout:0,4 con **AUC**: 0.6581. El AUC indica que la red tiene una excelente capacidad en distinguir las dos clases, aunque el valor de perdida en validación sugiere que el modelo no se está ajustando tan bien a los datos de validación como a los datos de entrenamiento.

Modelo 3 - Red neuronal Convolucional: Se establece para la primera capa convolucional 32 filtros y la segunda 64, dropout de 0,5, 150 épocas, kernel de 7x7 y se utiliza he_normal para una inicialización de pesos.

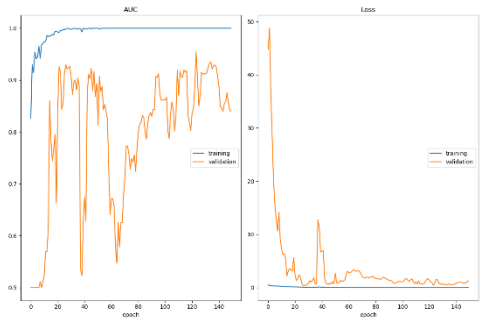

Figura 17. Resultados AUC – Modelo 3

Se obtiene en training un **AUC**: 1 y **pérdida**: 0,001 y en validación **AUC**: 0,955 y **pérdida**: 0,373; si bien los datos indican que probablemente el modelo esté sobreajustado, se logra la métrica en validación.

Modelo 4 – Red Neuronal Convolucional VGG16: este modelo utiliza pesos preentrenados de ImageNet y se excluye la capa de clasificación final. Las imágenes de entrada que el modelo espera tienen un tamaño de 224x224 píxeles con 3 canales de color (RGB).

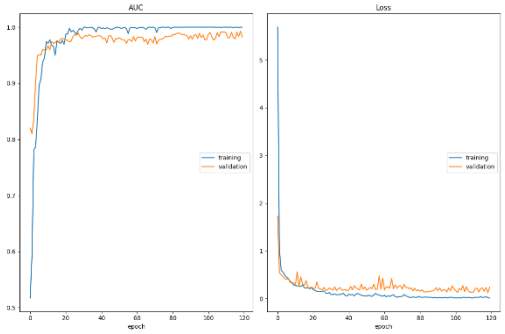

Figura 18. Resultados AUC – Modelo 4

Con 120 épocas, se obtuvo en training un **AUC**: 1 y **pérdida**: 0,013 y en validación **AUC**: 0,993 y **pérdida**: 0,121. El modelo permite distinguir entre clases con alta precisión. Se cumple con la métrica objetivo.

Modelo 5 – ResNet: este modelo utiliza pesos preentrenados de ResNet50, excluyendo la capa superior; se aplica una capa flatten y se añaden varias capas densas con activación ReLU. Con los siguientes Datos de Entrenamiento: (175, 224, 224, 3), Datos de Validación: (75, 224, 224, 3) y Épocas: 120.

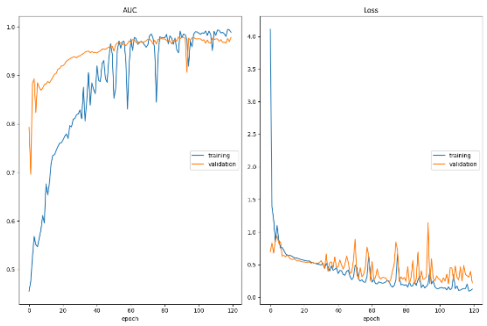

Figura 19. Resultados AUC – Modelo 5

Se obtuvo en training un **AUC**: 0,995 y **pérdida**: 0,094 y en validación **AUC**: 0,979 y **pérdida**: 0,205. El modelo tiene un rendimiento excelente tanto en entrenamiento como en validación, con AUCs cercanos a 1 y pérdidas bajas. Esto indica que el modelo está bien ajustado y tiene una buena capacidad de aprendizaje.

A continuación, se resumen los resultados de los modelos en los conjuntos entrenamiento y validación:

Modelo	Training AUC	ValidaciónAUC	TrainingLoss	ValidaciónLoss
Sencilla	0.8987	0.8851	0.3844	0.4371
Multicapa	0,960	0,964	0,262	0,287
Convolucional	1	0,955	0,001	0,373
VGG16	1	0,993	0,013	0,121
ResNet50	0,995	0,979	0,094	0,205

Tabla 1. AUC y pérdida de training y validación

De acuerdo con los resultados obtenidos, se observa que el modelo Red Neuronal Convolucional VGG16 obtuvo el mayor AUC tanto en entrenamiento (100%) como en validación (99,3%). Estos valores indican que el modelo fue capaz de clasificar correctamente la mayoría de las muestras en ambos conjuntos de datos y presenta la menor pérdida en validación. ResNet50 tiene el mejor balance entre rendimiento en validación y pérdida, lo que sugiere una excelente capacidad de aprendizaje. Por otro lado, el modelo de red neuronal sencilla tuvo un rendimiento inferior, aunque sus métricas son similares en entrenamiento (89,87%) y validación (88,51%) lo que le permite diferenciar bien entre las clases; igualmente sucede para el modelo multicapa donde las pérdidas en entrenamiento (0,262) y validación (0,287) son bajas y están muy cercas entre sí, lo cual indica baja probabilidad de sobreajuste.

De lo anterior se puede inferir que durante el desarrollo del proyecto se diseñaron y evaluaron cinco arquitecturas de

redes neuronales para la detección automática de frailejones en imágenes del páramo de Chingaza y Cruz verde. Todos los modelos planteados alcanzaron un desempeño satisfactorio, superando los umbrales establecidos para cada arquitectura, lo que demuestra su capacidad de detección precisa de frailejones.

Evaluación cualitativa:

La figura 20. corresponde a la predicción del número de frailejones sobre la imagen 'IMG_3451.JPG', para el desarrollo e inspección del modelo y su aplicación para reconocimiento de densidad poblacional de frailejones en el páramo de Chingaza y Cruz Verde. Los recuadros amarillos muestran la predicción de frailejón sobre la imagen general:

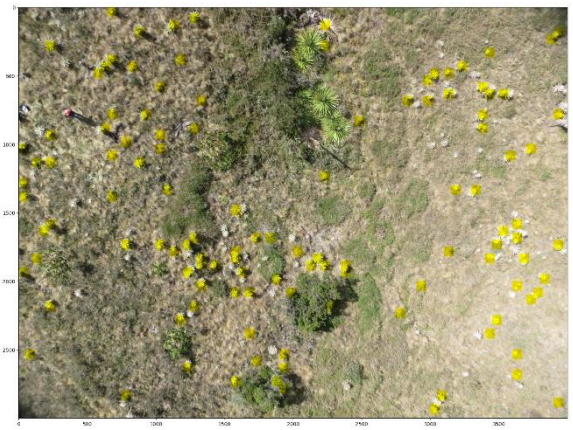

Figura 20. Predicción de frailejones sobre 'IMG_3451.JPG' con Modelo 1

En sentido general, se observa que el modelo es exitoso para determinar las zonas donde existen frailejones e identificarlos. Si bien no es exitoso en el 100% de los casos, se observan puntos amarillos en los lugares donde existen frailejones o en lugares cercanos, lo cual facilita el cálculo automático de la densidad poblacional de frailejones en el páramo por medio de reconocimiento de imágenes.

Cálculo de densidad poblacional:

El modelo seleccionado encontró satisfactoriamente 118 frailejones sobre la imagen de dimensiones 3000 x 4000 pixeles. Teniendo el valor del área relacionada a esta fotografía, se puede calcular la densidad poblacional en unidades/km^2. Para este ejercicio, calcularemos la densidad poblacional sobre el total de recuadros en la imagen.

Cada recuadro de color amarillo tiene tamaño 50 x 50 pixeles, por lo cual, la imagen tiene dimensiones de 60 x 80 recuadros. El total de recuadros es 4.800. Utilizando la fórmula para el cálculo de densidad poblacional, encontramos que:

Densidad poblacional = # frailejones (recuadros amarillos) / 4.800 recuadros totales.

Densidad poblacional = 118 frailejones / 4.800 recuadros.

Densidad poblacional = 0,0245 frailejones / recuadro.

Como conclusión del ejercicio, se demuestra de forma exitosa que el desarrollo de modelos de reconocimiento de imágenes es útil para detectar zonas de baja densidad poblacional, y de esta forma, que éstas sean objeto para estrategias de reforestación. Los metadatos de cada imagen contienen sus coordenadas espaciales por lo que las autoridades competentes ya pueden contar con herramientas de detección preventiva y automatizada. Lo anterior da respuesta al objetivo de negocio planteado originalmente.

RECOMENDACIONES DE NEGOCIO

Enmarcados en los objetivos de minería de datos y de negocio se recomiendan las siguientes estrategias según los modelamientos establecidos basados en redes neuronales:

- Implementar el modelo más eficiente (ResNet50 o VGG16) para detectar frailejones con alta precisión, priorizando un balance entre precisión y aprendizaje. Estos modelos en la detección permitirán identificar frailejones con una tasa de error mínima, garantizando una alta confiabilidad en la toma de decisiones de conservación.

- Desarrollar un sistema de monitoreo para la captura de imágenes a intervalos regulares, con el fin de evaluar las áreas críticas del páramo, mapear las poblaciones de frailejones y monitorear cambios a lo largo del tiempo, mediante la implementación de los modelos de detección y las imágenes recolectadas.

- Con los datos obtenidos a partir de la implementación de los modelos, crear un tablero de control que permita una visualización clara y en tiempo real de las zonas con baja densidad de frailejones, facilitando las intervenciones rápidas para restaurar áreas críticas y mejorar la toma de decisiones a nivel de conservación.

BIBLIOGRAFIA

- García-Llamas, S. L., Sarmiento, C., & Gutierrez, L. F. (2020). Deep learning for automatic identification of Espeletia species in the Colombian Andes. *Remote Sensing in Ecology and Conservation*, 6(3), 390-402.
- Ruiz-Olaya, A. F., Gómez-Ruiz, J. A., & Armenteras, D. (2018). Object-based image analysis for mapping frailejones (Espeletia spp.) in the Colombian Andes. *International Journal of Applied Earth Observation and Geoinformation*, 69, 152-162.
- Rodríguez, P. (2024, febrero 8). ¿Cómo aporta el desarrollo sostenible a la recuperación de paramos? Edu.co. https://virtual.cuc.edu.co/blog/c%C3%B3mo-aporta-el-desarrollo-sostenible-a-la-recuperaci%C3%B3n-de-paramos